开源时代

OPEN SOURCE TIMES

2009年1月刊 总第五期

开源业界：

Ubuntu用户应该关心Debian社区
2008开源软件的10大胜利
Sun为什么将会倒下？

CU技术沙龙：开源新时代&网络优化

2009年1月9日下午，ChinaUnix技术沙龙在新世纪饭店顺利结束。作为本次IT168年会的主要议程之一，本次沙龙无论从演讲嘉宾，还是听众人数和质量，都达到了一个新的高度。不仅有来自业内的专家领导，还有来自生产一线的技术领袖，整个会场讨论气氛热烈，嘉宾和网友的互动卓有成效。



社区扫描：

Ubuntu Mobile考虑使用Qt
热点回顾：Linux2.6.28五大特性
BSD发行版发布：FreeBSD 7.1

行业观察：

Windows 7真的能战胜Linux？
点评Linux难称完美的几大命门

技术新知：

Nagios监控系统配置实例
初探Cherokee：号称最快的Web服务器
独辟蹊径网络安装系列之Debian/Ubuntu

专家访谈：

专访内核专家Conke.hu

ChinaUnix.net



2009年1月刊 总第五期

编辑出品：ChinaUnix.net

网络发行：ChinaUnix

主编：江晖
技术主编：樊强
执行编辑：周荣茂
内容编辑：覃里　周荣茂
李倩　胡铭娅
技术委员会：高延斌　马路遥
白金
美术编辑：林在子
交流论坛：bbs.chinaunix.net
电子版下载：www.chinaunix.net
www.itpub.net
www.ixpub.net
联系我们：
qinli@staff.chinaunix.net
投稿邮箱：
qinli@staff.chinaunix.net

媒体支持：IT168.com

广告联系：温玉琴
电话：010-82658790
手机：13801339139
E-mail：wyq@it168.com

内容目录

卷首语 004

开源业界

2008年开源软件10大胜利 005
业界用户不使用Linux的八大原因 006
Sun什么时候会倒掉？ 008
Linux安全的未来 010
Intel开源操作系统Moblin 2开测 013
饮水思源：Ubuntu用户应关注Debian 014
Novell裁减不足100名员工 017
微软高层：GPL 协议机遇与挑战并存 017
PayPal的开源极具成本效益 018
提升Qt的应用 诺基亚即将增加授权选择 019

社区扫描

Ubuntu Mobile考虑使用Qt 021
Gentoo Portage包管理系统已支持Git 021
Python 3的演变 021
热点回顾:Linux 2.6.28 五大特性 022
Perl迁移至Git版本控制系统 022
BSD发行版发布：FreeBSD 7.1 022
FreeBSD 7.1:从Sun获取一些帮助 022
Mozilla向Ogg项目捐赠10万美元 024

专家专栏

Conke.hu：做面向Linux爱好者的开源项目 025

行业观察

企业如何监管开源软件使用情况？ 029
Linux真正价值不仅在于产品本身 030
关于国产软件、本国软件与开源软件的概念纷争 031
全面普及企业级Linux：仍重而道远 033
Windows 7能够战胜Linux吗？ 035
中国开源软件利用多数不规范 潜在风险巨大 036
点评Linux难称完美的几大命门 036

技术沙龙

IT168技术精英年会CU论坛回眸 040

技术新知

独辟蹊径网络安装系列之Debian/Ubuntu 042
初探Cherokee：号称最快的web服务器 055
nagios监控系统配置过程 060
使用SystemImager快速部署系统 079
Apache升级到了nginx的几个注意点 091
Linux系统下Bugzilla for Oracle 安装笔记 093

网友热评

097

版权声明
杂志内容来自ChinaUnix社区及互联网，电子杂志的宗旨是为了更好地传递开源最新自寻和技术经验。如有版权问题敬请联系，我们将会在第一时间做出处理。

致谢
本杂志得到ChinaUnix网站Linux时代社区版主的大力支持，技术文章大部分来自版主推荐，更多技术文章可以访问Linux时代精华区。
本刊分析评论部分文章来自IT168技术频道。

卷首语

2009 年第一期《开源时代》杂志今天终于和大家见面了，因为春节假期的缘故，本期杂志延迟了一周左右的时间，在此向大家说声抱歉！一年之计在于春，希望和大家一起，从 2009 年第一期《开源时代》开始，将我们的杂志做到更好，成为 ChinaUnix 网站为用户提供的又一个服务和展示平台。

和其他行业一样，年初是各行各业做计划的时候，在 IT 行业，没有几个预言是准确的，但是大家总是乐此不疲，因为这可以给大家带来美好的期望。在 Linux 和开源业界也是一样，正如一篇文章中写到的那样，2009 又将是 Linux 桌面之年？年复一年，大家总是在期待 Linux 桌面之年的到来，可是随着微软和 Apple 等竞争对手的咄咄逼人，Linux 桌面能否在未来几年中迎来她的辉煌？金融危机下的 IT 产业也是岌岌可危，但是对于 Linux 桌面来说，也许是个机遇。Linus 先生在 Gnome 和 KDE 桌面环境的选择中也给了我们启示，社区的多元化确实能给用户带来更多的选择，但统一标准的缺失更有可能是 Linux 桌面不能崛起的罪魁祸首。

作为技术社区，技术沙龙活动自然是我们展开工作的一个重点。2009 年 1 月 9 日，北京新世纪日航饭店，ChinaUnix 的两场技术论坛---《开源新时代》和《网络优化》如期举行，作为 IT168 技术精英年会的主要技术日程，有超过 400 人的 ChinaUnix 网友报名参与了这两个技术论坛活动。有来自社区、业界、政府部门的多位技术领袖、企业领导和政府官员，参与了我们的圆桌会议、现场问答、技术讲座等活动，得到了现场观众和网友朋友的一致好评。

作为 ChinaUnix 论坛的新开设版面，开源项目孵化版也在 2009 年年初和大家见面了。“中国特色”说多了，有的时候自己都感觉到变味。传统的开源项目孵化网站也没有像我们这样运作的，依托于论坛，大家更多的还是依赖邮件列表和 SF 这样的网站。但是作为我们社区对国内开源软件事业的支持，我们希望这个版面能够吸引和培养一批优秀的开源软件爱好者，以论坛这种有中国特色的交流方式，来打造我们的开源软件项目，最终有一天能够得到国内外开源同仁们的认同。

开源社区的生存和发展完全依托于开源用户的，《开源时代》杂志也是一样，更多的意见和发展建议还需要大家来提出。如果您对我们杂志有什么好的想法，或者我们有做的不足的地方，无论是内容编排还是杂志发行等方面，都欢迎您提出！我的联系方：rmzhou@staff.chinaunix.net。

再次恭祝大家，牛年好运，工作顺心！

周荣茂

开源业界

2008 年开源软件 10 大胜利

在过去的一年中，Linux 操作系统和开源软件技术取得了令人振奋的进步。其中包括开源软件更加完善，Linux 内核已经逐步进入主流产品，包括机顶盒、移动电话。以下就是 10 大开源软件的胜利。

1、Mozilla 的 Firefox 3 发布

Mozilla 的 Firefox 网页浏览器是最成功的开源软件应用，这款浏览器在今年正式推出了 Firefox 3，并在发布第一天创造了 800 万下载量。

2、Google 公布 Chrome

Google 去年放出了浏览器重磅炸弹，Chrome 浏览器，这个基于开源 WebKit 核心的浏览器支持 Windows 操作系统，同时融合了创新性的 技术，包括多进程和内置的任务管理器，用户可以查看每个进程的资源消耗状况，并关闭单独的标签。Chrome 在上个月结束 Beta 测试，并将很快出现在厂 商品牌机的预装程序中。

3、手机迎来 Android 平台

Google 去年的另外一个重要举措是，推出基于 Linux 操作系统的 Android 开源手机平台。虽然旗舰版 G1 手机在发布之初略微有些令人失望。不过 Google 的开源态度决定了这款平台的长期发展。





4、KDE 4 发布

Linux 桌面环境在去年升级至 KDE 4，这个重要更新彻底改变了开源桌面环境的。最初推出的 4.0 版本稍显粗糙，但是随后更新的 4.1 环境则彻底解决了问题。预计 KDE 4.2 环境将在 1 月份推出，用户体验将得到增强。

5、诺基亚解放 Symbian

移动电话巨头诺基亚已经成为了开源软件中具有影响力的一员，因为该公司去年重新调整了战略，把 Symbian 平台作为开源平台推出。Symbian 将从诺基亚中解放出来，并最终按照 Eclipse 公

共许可协议分发，并逐渐走向社区开发的道路。

6、诺基亚收购 Trolltech

在诺基亚收购 Trolltech 之后，这个开源 QT 工具包将成为移动桌面开发之间的桥梁，通过 Qt 工具包开发的程序，可以支持几个不同的移动平台，Qt 现在已经增加了 Symbian S60 平台支持。

7、Maemo 5 发布

Maemo 5 是基于 Linux 的互联网 Tablet PC 操作系统，据诺基亚 Ari Jaaksi 表示，Maemo 5 提供了令人兴奋的技术改进和用户界面，并已经公布了 SDK，这种操作系统可能会成为今后主流的移动互联网设备及 Tablet Pc 的操作系统。

8、Python 3 预览版放出

开源开发语言在去年也取得了进步，Python 3 编程语言向后兼容，并支持强大的新功能和语法。另外，Google App 引擎也将支持 Python 3 语言。

9、Mono 2 开源平台公布

Mono 2 是一个开源版本的微软.net 平台，目前的版本是 2.0，它支持 C# 3.0，也同时支持 Mac OSX 操作系统，这位曾经在 Windows 下编程的程序员提供了便利。另外，去年官方公布了 MonoDevelop 1.0 开发环境，用于建立基于 GTK#的应用程序。据悉该厂商将在今年为 Linux 用户提供开源版的 SliverLight 网页插件支持。

10、Sun 公司的 OpenSolaris

去年 Sun 发布了首个 OpenSolaris 操作系统，这个平台的目的是建立一个围绕 Solaris 操作系统相关技术的开发者社区。该项目针对的对象包括致力于开发和改进 Solaris 操作系统。

业界用户不使用 Linux 的八大原因

现在，虽然 Linux 技术发展和普及的势头很迅猛，不过，仍然有一些企业坚持不使用 Linux，他们的理由也很充分。以下就列举了其中的 8 个原因。

1、专利侵权

使用 Linux 操作系统极有可能会使用户在将来陷入专利侵权问题的牢笼，给用户带来无法量化的损失。比如微软就因为这个问题起诉过 Linux 厂商 Novell 和 SuSE。

“如果某个企业用户想要部署 Linux，而不是 Windows 或者其他专有的 Unix 系统，那么这就将是一个潜在的风险，” 企业管理协会（EMA）资深分析师 Andi Mann 说。EMA 是一个独立的行业分析机构并且专门从事 IT 管理市场的咨询公司。

这种潜在风险造成的成本是不是能够足以使企业用户放弃使用 Linux 的念头呢?

2、Linux 许可有可能成本更高

市场分析机构 Yankee Group 资深分析师 Laura DiDio 在自己的研究报告“微软、Sun 反击 Linux，力争让操作系统成本更低”中指出，微软和 Sun 的专有操作系统产品的成本可能要比红帽和 Novell 的 Linux 产品成本低大约 10%到 100%，这取决于操作系统的配置和许可合同的长短。DiDio 的结论比较适用于 Windows Server 2003，Solaris 10 以及 Red Hat Enterprise Linux 和 SuSE Linux 。

3、内置/捆绑功能够大幅度节省资金

虽然，大多数 Linux 发行版也自带很多免费软件。但是，你花在专有操作系统上的钱绝对也令你物有所值---你可以得到数量非常众多、功能齐全的各种工具和服务。

DiDio 指出，“如果你买了微软的软件保证（Software Assurance），你就需要支付一定的保费，但你同时也得到了很多东西。你可以获得免费培训。而且还能访问在线资产管理来跟踪许可，这样你就能避免买 多或买少，这给你带来了巨大的方便。并且在家庭使用权方面你也可以获得较大的折扣。”

DiDio 还表示，微软的软件保证即使对于只有 10 几个员工的小企业也绝对是物有所值，能够为它们节省数千美元，“而大型企业又可能会节省数百万美元。”

此外，DiDio 还补充说：“经过改进的 Windows 补丁管理工具能够使网络管理员的工作效率大大提高---平均百分之五十至百分之八十左右，并且 不会造成成本的增加。并且微软免费提供 Windows 服务器更新服务（WSUS）。而如果你选择 Linux 操作系统，那么你不得不将 IT 预算的百分之二十 至百分之二十五用来购买第三方管理和性能优化工具，而这些工具已经包含在 Windows 许可中了。”

需要补充的是，Windows 并不是唯一的 Linux 替代品。

如果你打算在服务器上使用虚拟化或类虚拟化的环境，“VMware 的虚拟化技术往往会使每个服务器的每个 CPU 的成本增加一千美元或更多，” Sun 公司 Solaris 营销部副总裁 Marc Hamilton 说。Hamilton 还指出，包括 Sun 的 Containers 虚拟化设施不会增加任何额外费用。

4、Linux 维护成本高

就购买成本而言， Linux 操作系统通常会比专有操作系统要便宜。EMA 的 Mann 在其“Linux 管理的真相”中指出，Linux 的购买费用---不包括支持费用--- 通常只有 Windows 的百分之十左右。例如，“对于一个有 100 个用户的网络应用环境，部署一个完全的 Windows 环境需要 6 万美元，而相应的开源软 件只需要 6 千美元。”

但是，这并没有包括管理工具的成本。“在我看来，纯微软环境更易于管理，” Mann 说。“这就是为什么比需要花大价钱购买 Linux 系统管理工具，没有管理工具，Linux 环境是很难管理的，这是大家都公认的。”

如果你不擅长管理服务器环境或者如果你没有合适的管理工具，Windows 将是一个更容易管理的系统而且成本也更低，Mann 说。“软件发行管理、补 丁管理……非 Linux 环境有很多更加先进的管理工具，特别是 Windows 操作系统，同时还有 Unix 和 Solaris 。”

Linux 和 Windows 管理工具的成本从零到几十万美元不等，比如 CA Unicenter 或 IBM 公司出品的 Tivoli。“但是，对于小型环境而言，比较便宜的工具完全能够胜任，” Mann 说。

5、聘请 Linux 技术人员可能需要花费更多的钱

“现在，操作系统治持维护人员的知识遂平有一个显著的差异，” 资深 IT 管理人员 Chris de Herrera 说。“Windows 支持所需要的专业知识比较少，想比之下， Linux 则需要很多。”

Herrera 说：“Windows 的每一个应用程序都是已经预编译好的，所以一旦它们安装完毕，一般都能正常运行。而对于 Linux 操作系统而言， 情况可能并非如此。另外再加上一些特殊要求，比如 Linux 技术人员需要理解源代码管理和版本，一个例子就是 DLL 地狱。有些 Linux 应用程序需要一

定版本的类库才能运行，这就有可能与其它应用程序冲突。”

Herrera 表示：“系统支持和维护人员经验的差异以初级开发人员为界，而不仅仅是做系统管理。另外，Linux 安装升级的过程可能会有些复杂，这取决于使用的升级方法。如果你选择的应用不支持，那么你必须手动完成升级。”

“企业使用 Linux 的前提条件是自己的 IT 员工对于 Linux 集成和基础设施必须相当熟悉，” ExtremeLabs 公司总经理 Tom Henderson 说。

“如果你的 IT 技术人员对于 Linux 不是非常熟悉，那么你就需要对工作人员进行培训或者招聘一些具备这些技术的员工，” Henderson 说。“经验丰富的 Unix/Linux/HP-UX/Solaris 专业人员聘用价格往往那个非常高。”

6、使用非 Linux 的 Unix 可能是一个更好的选择

Linux 不是唯一能够运行在 X86 硬件上的非 Windows 操作系统，比如 Sun 公司的 Solaris。这点很容易忘记。“与 Linux 发行版相比，Solaris 的支持费用更低，”Hamilton 说。“比如，Solaris 10 的支持成本通常要比 Red Hat Linux 低百分之十至百分之五十或更多。”

造成这种现象的一个原因就是 Solaris 具有的一些特定功能，比如 Solaris 故障管理架构。如果出现一个内存 DIMM 失败，“Solaris 10 操作系统将自动重新配置，而不需要重新启动。”虽然内存 DIMM 失败并不经常发生，“可是，如果你的网站有 100 或 1000 台服务器，那么你就不得不考虑这个问题，” Hamilton 说。例如，如果你的远距离数据中心采用 Linux 操作系统，并且有 200 台服务器，“这可能意味着你将每天都要派出技术人员修复这个错误。”

7、硬件支持问题

“Linux 对于硬件和驱动器的支持问题也应该引起注意，而 Windows 通常没有这方面的问题”Mann 说。

“硬件兼容性问题并不是 Linux 独有的，”开源分析公司 Red Monk 首席分析师 Stephen O'Grady 指出。 Linux 对于硬件的支持正在变得越来越好。但是仍然存在一些问题，尤其是在桌面上。

“Linux 有可能缺少最常见设备的驱动支持，比如信用卡刷卡机或销售点终端机，这可能会让你额外支付很大一笔费用。” IndependentAssessment 分析师 Alan Radding 说。

8、向 Linux 迁移成本过高

当然，Linux 的总体拥有成本当然还包括迁移成本。“关键应用的转换成本可能会很高，比如 Microsoft Exchange Server，因为建立在 Exchange 之上的电子邮件、服务和应用都需要更换、修改或修复，” Henderson 指出。

应用可用性也是一个值得关注的问题，O'Grady 指出。“有很多应用软件只支持 Windows 服务器或桌面。比如，像 QuickBooks 这样实用的软件却不能运行在 Linux 平台上。”

Sun 什么时候会倒掉？

我不是在幸灾乐祸的看着 sun 亏损年报。我想分析一下，请大家拍砖。

举个例子，假设 Microsoft、IBM、Intel、Sun 和 HP 这些公司都是经营商店/超市的。那么 Microsoft 就是沃尔玛，Intel 是肯德基，AMD 是超市发，虽然沃尔玛和肯德基也会卖高价的东西，

但是基本都是便宜货，销量大，总利润无人能敌。AMD 呢，和 Intel 有竞争，也有不同的路线，盈利点有重叠也有不同。IBM 是国美电器，卖的东西贼多贼贵(比苏宁的贵)，HP 是苏宁电器，名头没有国美大， 但是比国美实惠点。而 Sun 就是一家高档手机专卖店(服务器卖的真是贵呀)，专门卖给成功人士。



那么好了，经济危机来了。人们可以不买奢侈品，但是吃的得买吧！于是 Microsoft 开始搞让利活动(office199 一套)，大卖特卖，利润少了点，但是日子还能过。intel 利润下降了，开始裁减一些卖场售货员，提高效率。HP 也受了影响，开始减少推销人员，不过九阳豆浆机卖得很好日子也能过。IBM 呢，因为东西种类多，东方不亮西方亮，电视机销量下跌了，电饭锅却多卖了好多。

最惨的就是 Sun 了，它坚持高价高质量，只卖给成功人士。还记得 2000 年的辉煌吧，多少成功人士办的公司都要购买 Sun 的奢侈品，Sun 那个火呀， 一度不把卖牙膏和牙刷的 Microsoft、Intel 放在眼里：股票市值还不如我呢。结果后来大家裤腰带子一紧，奢侈品就被拿出来典当了。突然冒出来一个叫狗狗(Google)的有钱人，需要很多手机，Sun 说，我的手机好啊，显得你是成功人士！狗狗说，切，我自己 diy---然后找了 Intel 和一个叫 Dell 什么的穷鬼开始自己捣鼓，刷机系统用了红帽的山寨版，据说还挺好用。

再后来，Sun 看着自己的奢侈手机买不动了，于是说，以后买手机免费刷机，我这个刷机系统还能在低端手机上用呢！大家说，你那个臃肿的刷机系统谁用啊！Sun 急了：大家都来免费用用看看(solaris10 开源)！大家仍然不理不睬，但是一些路过的学生进去尝试了一下。更后来，Microsoft 和 Intel 的卖场，HP/IBM 的卖场纷纷推出性价比更好的手机，据说功能款式一点不差，价钱有的还超便宜，卖得那个叫火呀。Sun 更妒 嫉了，"成功人士都买我的产品"！然后还在自己的网站上大打广告。路过的人一看：咳，还不是当年 moto 的 V3，样子和颜色改了改，还想像当年那样卖 6000 来块钱？做梦呢。然后 08 年的经济危机又来了，Sun 的奢侈品没卖出去多少----成功人士可以不买奢侈的手机，但是牙膏和牙刷还是要去 Microsoft 那里继续买的，虽然也是省着用，不多买。

还有一根稻草，一个叫 linux 的穷鬼和一个叫 bsd 的穷鬼在 Sun 的商店旁边开了两家水货手机/山寨机专卖店，保修差了些，不过货真的很好，总拥有价钱是 Sun 那家店的几十分之一。好多人都去买。山寨货的销路也相当不错，好多人都是冲着大功率外放去的。

你说 Sun 这家店，还能开下去么。

Linux 安全的未来

不要仅仅因为 Linux 比其他操作系统更安全就认为你的 Linux 系统是安全的，开发人员和发行商将来能为系统管理员提供什么帮助呢?

你知道我已经写这个专栏有五年的时间了吗？在这五年之中都发生了些什么呢，我们看到 linux 的竞争者已经接受了它，而且 linux 已经可以作为一个桌面平台。

在 linux 安全领域，也有值得注意的进展，linux 的防火墙现在已经非常成熟，大量的嵌入式防火墙设备都基于它，与安全无关的设备也大量地采用了 linux，linux 支持令人吃惊的数量众多的安全工具，使得它称为安全审计员和安全顾问最喜爱的系统，此外，linux 已经形成了基于角色的访问控制操作系统，最著名的就是 NSA 的 SELinux。



但是 linux 安全的未来是怎么样的呢？我已经写了许多有关 linux 安全的现状和过去，但是还没有写过有关未来的东西，这个月，我将我想到的 linux 安全将走向何方以及它应该走向何方做一下总结。

现在有什么错误吗?

近来显示大量的人开始关注 linux 的安全并没有比微软的 windows 操作系统安全得多，在开始交火讨论之前，让我们先解释一下这个观点，首先，作 为个人来说，我认为 linux 是比 windows 更安全的，我已经在这个专栏的文章中反复重复过，用户在 linux 下比在 windows 下更容易控制他们的系统。

问题是 linux 用户，与 windows 用户类似，倾向于将他们的精力集中在让系统做他们想让系统做的事情上，他们太信任系统内置或默认的安全设置，那么，当不可避免的软件 bug 出现时，那些 bug 的影响趋于更广泛，比起预防来说要做的工作就更多了。

例如：如果我们运行 BINDv9 提供名称解析服务，将花一些工作和研究才能使其工作起来，要将其放在一个 chroot 环境中需要更多的努 力，chroot 可以将 named 进程放在文件系统的一个子集中运行，因此，当一个 BIND 漏洞被发现，大多数 BIND 用户可能都经历过没有放在 chroot 环境的痛苦，如果运行的是微软的名称解析服务—它没有 BIND 那么多的安全特征，那可能痛苦指数是一样的。

所有这一切都是要简单地告诉你大部分 linux 安全特征和功能并没有让 linux 用户受益，结果是，至少按照我朋友所说的进行专业的渗透测试，攻破你普通的 Redhat 企业版并不比普通的 windows2003 系统困难。

这是不幸的也是让人非常吃惊的，它的代码是完全透明的，linux 仍然有类似的软件 bug，一般来说和 windows 的数量和频率几乎一样。和 windows 一样，linux 是由成百上千的人开发出来的一大堆复杂的代码，代码越多，bug 可能就会越多，不是吗?

我最近接受 SearchSecurity.com 采访时谈到了一篇关于微软研究基金指导的安全革新文章，研究结果表明 windows 比 linux 更安全，结论主要基于常见安全 bug 和发行补丁的平均时间，我相信我是正确地评论了研究结果，不考虑 linux 的其他安全优势，如定制能力和软件包的选择，换句话说，我感觉这个研究更多的是比较默认安装情景，而不考虑每个操作系统被它的用户进行安全加固的因素。

但是我考虑得更多，我更担心或许一个平台的安全隐患是不能统计的，除非大多数系统运行的平台实际上触发了隐患，严格来说这不是一个终端用户行为的作用，我也不会指责系统管理员，因为我在后面会详述，我想 linux 开发人员和发行商必需继续想出让安全特征更普通、透明和更容易配置和使用的方法，顺便说 一下，因为我正在比较 linux 和 windows，公平起见我应该指出 windows 也有许多安全特征，但用户也很少使用它们。

好，linux 和 windwos 在默认情况下它们都不安全，在软件 bug 和安全补丁两个都不相上下。

两个操作系统都是使用简单的任意访问控制模式来进行安全设置，在这个模式下，一个超级用户账号—在 linux 下是 root，在 windows 下是 Administrator—拥有控制全部系统的权力，包括其他用户的文件，在这两个操作系统中，组成员可以被用来创建不同等级的访问，比如说，root 可以进行多种授权，实际上，在大多数系统上你不得不作为特权用户登陆或临时变成那个用户为了完成重要的事情。

结果，任何用特权用户运行的进程能完全控制 linux 或 windwos 系统，但是，我是作为非特权用户登陆配置我重要的后台进程的，这些后台进程出现 了 bug 是不会影响到整个系统的。但是其他软件的 bug 可能使得它从一个非 root 进程升级它的权限，例如：假设你已经获得了一个运行 Apache 的 web 服务器，一天一个攻击者控制攻击程序攻击一个没打补丁的 Apache 缓冲区溢出漏洞，结果是攻击者在你的服务器上获得了一个 shell 会话，从这 一点来说，攻击者正作为 www 运行，因为 Apache 是作为 www 运行的，假设这个系统还有一个未打补丁的内核漏洞，那将导致本地权限提升。

系统管理员可能已经知道了这个漏洞，但是补丁还没有出来，毕竟，严格地说是一个本地漏洞，除你之外没有人在这个系统上有 shell 权限，谁想在给内核打补丁后不得不重新启动呢? 但是现在一个远程攻击者有了一个本地 shell 访问权限，如果他成功地利用了这个内核漏洞，他就是 root 了，这就是常见的入侵场景，但是使用了 root-takes-all 安全模块后不用再担心这个了。

这就是 linux 安全的现状，保护 linux 需要我们花费相当多的努力利用复杂的安全特征，这些复杂的特征默认情况下往往没有启用，要保持所有补丁都是最新的。我们在一个好的公司里：大多数使用现代操作系统它们拥有相同的局限和挑战。

强制访问控制

我已经提到在 linux、UNIX 和 windows 上的访问控制或文件权限是任意的，这就是一个弱的安全模块，那么，SELinux 怎么养呢? 它使用基于角色的访问控制(RBAC)和类型增强(TE)，它们两个都是强制访问控制的实例，是的，的确，它就是这样。但是我担心这或许不是 linux 安 全的未来，原因是 SELinux 不是当前 linux 安全的重要组成部分。

RBAC 限制用户的行为和对系统资源的访问，基于仔细定义的规则，这些规则比常见的 UNIX 组

机制相似但更深远，类似地，类型增强限制进程的活动，基 于它们预先定义的操作域，RBAC 和 TE 对网络的影响是创建将用户和进程操作分开的竖井(我的术语)，严格地限制竖井之间的交互。

这是一个非常优雅的有效的安全模型，但是，对于大多数人们来说，RBAC、TE 和其他强制访问控制太复杂，很难管理。大多数人看来，SELinux 和 类似的操作系统命中注定只能用于某些特定领域：操作系统对于需要它的人来说非常有用但注定不能被广泛接受。不管是赞美 SELinux 的安全架构还是对 RBAC 原理的迷恋，我认为强制访问控制本身并不是 Linux 安全的革命。

底层管理程序和虚拟机

如果 RBAC 和 TE 事实证明太难以使用不能很好地在操作系统层区分安全破坏，底层管理程序和虚拟机(vm)可能在一个更高层次实现这个目标，我们已经 熟悉虚拟机处于两个不同的上下文中：运行时虚拟环境(就象那些使用 java 的程序)和虚拟平台(如 Vmware、plex86 和 VirtualPC，它运行你在一个虚拟的硬件环境中运行整个操作系统)。

java 虚拟机被设计成有特殊的安全特征，最值得注意的就是 java 沙箱，通常，java 安全来自于 java 小程序运行时与真实的操作系统资源是隔开 的，每一个事情都是通过 java 虚拟机完成的，除了一个好的安全模型之外，对于程序员和最终用户在使用上也相对简单安全些，正是因为这些原因 java 已经受到普遍应用。

底层管理程序隔离运行在相同硬件上的虚拟机，限制它们交互和防止安全破坏，IBM 已经为底层管理程序创建了一个叫做 sHype 的安全架构，一个开源的底层管理程序/虚拟机项目—叫做 Xen —到现在也是可用的了。

底层管理程序的目的是为了防止一个虚拟机与其他运行在同一硬件上的虚拟机发生冲突，例如：独占共享资源，有一些智能管理系统在这个层次已经做得很强大了，或许还有一些潜在的隐患，至少，增加了传统的入侵检测系统(IDS)检查系统泄密的复杂性。

强制访问控制和底层管理程序/虚拟机不是相互排斥的，一方面，我的主张是，强化朋友和伙伴对安全分析的重视，底层管理程序比 MAC 在 linux 安全未 来的发展更具有潜力，但是另一方面，这两者可以结合起来使用，设想一下一个大型的、强大的服务器系统运行几个由底层管理程序控制的虚拟机的情景，一个 VM 能运行在一个通用的操作系统上，如 linux，提供 web 服务，另一个 VM 为敏感信息提供数据库服务，它们都能运行在一个基于 MAC 的操作系统上，如 SELinux，两个 VM 都能从强制安全控制受益，使用 SELinux 能提供一层额外的安全保护。

基于异常的入侵检测和防病毒系统

如 MAC 和底层管理程序目前已经生根发芽了，但是将来可能还会出现更大的影响力的技术：如基于异常的入侵检测系统。基于不规则的 IDS 很简单：它包括创建一个正常网络或系统活动的基线，以及在任何突发意外或有反常活动被检查到时发送警告。

基于特征的系统的致命弱点是如果攻击方式是最新的，那么在你的 IDS 的特征数据库中就要有与之对应的特征，如果没有就检测不到。

使用基于异常的 IDS，与之相反，任何新的攻击方式只要不是正常行为都能被检查到，IDS 管理员必须训练和周期地重新训练 IDS 系统，目的是创建一个正常行为的基线，这会导致一段时间内频繁地出现虚报现象，直到这个基线被调试好为止。

1999 年我出席了由 Marcus Ranum 主持的一个讲座，他提到了基于异常的系统是 IDS 未来的发展方向，象这样的产品已经可用，如来自 Lancope 和 Arbor Network 的产品，但是我仍然希望

有人能想出如何才能开发出便宜的易用的东西，这可能导致一类网络底层管理程序增加同样的智能到网络上，无论是虚拟还是真实机器的构成，底层管理需都要借用虚拟平台。

顺便说一下，病毒扫描程序也需要从异常检测技术(如 IDS 一样)受益，这个观点现在已经被大多数病毒扫描程序证实，它们主要依赖于病毒特征的匹配，尽管如此还是要遭受大量的病毒/木马/蠕虫的爆发，目前基于病毒特征匹配的杀毒工具很明显基本上没什么效果了。

结论

前面这些就是我想到的 linux 安全的未来，同时，继续保持使用这个专栏中提到的技术：防火墙、病毒扫描程序、自动补丁/升级工具、VPN 和特殊应用程序安全控制如 chroot 以及审核跟踪。

那么，再见了，本专栏的文章全部结束，我将集中精力在其他新鲜事物上，我将继续我安全编辑的角色，将一如既往地支持 Linux journal，给大家带来一些安全领域的内容。

Intel 开源操作系统 Moblin 2 开测



Intel 为自家 Atom 平台上网本、MID 推出的开源操作系统 Moblin 虽然至今还未曾获得过公众的广泛关注，但 Intel 在这一项目上的开发却从未停止。日前，他们终于推出了第二代 Moblin 平台的首个 Alpha 测试版本。

Moblin 项目始于 2007 年，Intel 的目标是建立一个轻量级的开源操作系统平台，扩展 Atom 硬件平台的应用前景。目前，已经有 Linpus、GoS、Mandriva 等 Linux 厂商表态将推出基于 Moblin 核心的上网本专用 Linux 版本。

事实上 Moblin 2 本身就是 FedoraLinux 操作系统的修改版本。Intel 为适应上网本、MID 系统的应用环境，对其进行了多项优化，包括提高启动速度，专用的网络连接管理器等。用户界面上，它没有使用资源消耗量大的 GNOME 和 KDE，而是使用了一款轻量级的 Xfce 桌面环境。不过 Intel 表示，最终正式版的 Moblin2 会放弃 Xfce，换用功能更丰富，更适合移动设备的 Clutter 桌面环境。Clutter 来自于 Intel 去年收购的 OpenedHand 公司。

Moblin 2 的首个 Alpha 版本已经通过了宏碁 Aspire One 和戴尔 Inspiron Mini9 平台的兼容性测试，也可以安装在华硕 Eee PC 901 上，只不过无法使用无线网络模块。想要尝鲜的用户可以直接下载 Moblin2 的安装镜像，刻盘或拷贝的 U 盘上安装使用。

Moblin 2 Alpha 版下载页面：
http://moblin.org/documentation/getting-started-guides/test-drive-moblin

饮水思源：Ubuntu 用户应关注 Debian

当今最热门的桌面 Linux 非 Ubuntu 莫属，它已经拥有数量众多的爱好者，已经成为 PC 用户的主流操作系统之一。尽管 Ubuntu 相对还比较年轻，但 是它的前身却是拥有多年历史和经验的 Debian 项目，或者说它继承了 Debian 这个最古老、最具影响力的 Linux 发行版的优点。这两个最受欢迎的 Linux 发行版之间存在着密不可分的“血缘”关系。

尽管 Debian 对 Ubuntu 的过去和将来都有着重要的影响，不过很多新 Ubuntu 用户却对 Ubuntu 的根源知之甚少，自然也不知道它与 Debian 之间的关系。本篇文章将重点介绍 Debian 对 Ubuntu 的重要性，以及 Ubuntu 在 Debian 生态系统中所扮演的重要角色。



Debian：一个坚实强大的基石

Debian 由 Ian Murdock 在 1993 年创建，相比竞争对手而言，Debian 具有几大重要优势。它具有一个极其强大的包管理系统和丰富的可用软件。根据最近的统计数 据显示，在其主程序库中，仅仅稳定版的软件包就超过了 20000 个，而最新的不稳定开发版则有 30000 多个。

让 debian 区别于其它 Linux 发行版的另一个关键之处是，它具有一个非常完美的面向社区开发模式。Debian 是厂商中立的，其标准透明，无官僚作风。

尽管 Debian 具有非常强大的优势，不过它也有严重的弱点，尤其是桌面系统方面。它的开发周期过于冗长，每隔两到三年才发布一个版本，对于想体验最新软件的用户来说，这无疑是一个噩梦。

Debian 分三种形式提供：稳定版、测试版和不稳定版。稳定版被广泛用于服务器上，但是很多桌面用户更喜欢选择测试版或不稳定版，因为在这两个版本上可以使用当前比较流行的软件。不幸的是，这两个版本不能保障功能完整，且用户时常会碰到严重的异常错误。

Debian 项目的底层意识形态常常又阻碍了其发展。该项目崇尚民主，但缺乏一个强大的中心领导者，这导致了其决策过程的低效，且缺乏一个稳定的方向。开发工作通常会被一些特定问题的非技术争论而影响进度。

在过去几年中，Debian 管理模式的一些分歧已经大大减少了其参与者，迫使某些高水平的贡献

者不得已放弃该项目。尽管其它 Linux 发行版也同样存在此类问题，但 Debian 社区中此问题显得尤为突出。

Ubuntu：弥补不足成就伟业



Ubuntu Linux 最早于 2004 年作为 Debian 的一个分支出现，其创始人是南非企业家 Mark Shuttleworth。Ubuntu 项目由 Shuttleworth 的公司 Canonical 和社区志愿开发者共同努力开发而成，目的是实现一个现代 版的 Linux 版本，使其在桌面系统上真正具有竞争力，更适合主流非技术用户使用。

Ubuntu 的重点在于提高易用性，并且坚持定时发布新版本，即每隔六个月发布一个新版本。这确保了用户不再使用过时的软件。其发布计划一般是紧随桌面环境 GNOME 项目，Ubuntu 一般是在新版 GNOME 推出新版一个月后也推出新版。

Ubuntu 不是 Debian 的一次性分支。这两个 Linux 版本依然紧密联系，大量的 Ubuntu 软件包依然直接来自于上游的 Debian。在每个 Ubuntu 版本开发初期，新的更新软件包自动从 Debian 不稳定版本直接导入到 Ubuntu 中。导入完毕后，这些软件包在 Ubuntu 中被进行充分的测试和完善，以确保在正式版推出后具有高度的可靠性和健壮性。

从与 Debian 的关系来看，你可以把 Ubuntu 看做一系列 Debian 不稳定版的快照，不过它已经被进行了完善，而且提高了易用性。Ubuntu 开发者所做的工作是，在 Debian 的基础上简化软件安装过程，使其更易于被桌面计算机所使用。

Ubuntu 具有一个用户友好的 Live CD 盘，它具有一个现成的完整 Linux 环境，无需用户选择单独的的软件包或进行其它的选择，即可体验 Ubuntu 系统。Ubuntu 还包含一些辅助工 具，让你可以轻松安装和配置专有硬件驱动、编码器和其它用户需要的组件。这些工具使得 Ubuntu 只需经过很少修改或配置，就可以成为适合绝大多数环境的全功能版桌面平台。

Ubuntu 在很多方面具有重大改进，不过 Debian 也有自己的优势。Ubuntu 定时发布管理战略的劣势之一是，有时候不能包含一些有用的升级软 件包，而 Debian 不稳定版尽管不提供稳定性保障，但它的滚动更新却可以确保它总能提供最新的软件包。Debian 的另一个巨大优势是，它支持多种架构，而 Ubuntu 官方只支持标准的桌面计算机和 ARM 架构。

Ubuntu 与 Debian 的冲突

在很多方面，Ubuntu 对 Debian 的依赖依然非常严重，但是它们之间的关系却并非总是那么和睦。Ubuntu 的成功使其与 Debian 的关系开始变得紧张，人们指责 Ubuntu 是寄生虫，没有对 Debian 作出等价的回报。在过去数年中，Ubuntu 社区已经采取了重要的措施来缓解这种紧张关系，解决两者之间的冲突。

在 2005 年两者之间的争论第一次出现，Debian 创始人 Ian Murdock 表示，Debian 和 Ubuntu 要想实现双赢，需要保持这两个 Linux 版本之间的兼容性和互通性。

他表示，“如果 Ubuntu 是 Debian 家庭中的一员，它的成功就代表着 Debian 家庭的成功。但是，前提是 Ubuntu 还是 Debian 家族的 好儿子。我的担心是，它正在显现出变为任性孩子的迹象，早期的成功使其变得自大，开始怀疑自己是否还需要它的父亲。如果 Debian 能从 Ubuntu 中获益，两者之间的血脉将得以延续，Ubuntu 必须更强大，而不是简单的模仿 Debian。”

Shuttleworth 对 Murdock 的大部分观点表示赞同，强调协作是两者向前发展的唯一道路。

Shuttleworth 在几年前表示，“我认为，没有 Debian 就没有 Ubuntu。因此我认为 Ubuntu 是 Debian 一个扩展产品，Ubuntu 有责任和义务推动 Debian 的发展，因为 Ubuntu 的成功是因为站在伟大 Debian 的肩膀上。”

但是，它们两个在几个关键技术问题上存在意见分歧，例如在解决互操作性问题上两者就有不同的观点。Murdock 在 DCC 联盟担任要职，该组织由 Debian 和几个衍生 Linux 联合创建，旨在为 Debian 系统创建一个兼容 LSB 的共同核心。Murdock 希望这样可以保证所有基于 Debian 的 Linux 系统之间的兼容，防止发生他所担心的各自为政的现象。该计划在几个层面上都存在失误之处，其技术上的失败最终导致了它的瓦解。 Shuttleworth 曾准确的预言了这些问题，并且没有遵循 DCC 标准。

曾经有几个活动来让这两个社区增加协作，但是这些活动一直未取得较大成功。在 Debian 方面，建立了 Utnubu 项目来帮助将 Ubuntu 软件包加 入到 Debian 中。在 Ubuntu 方面，则创建了 Debian 协作团队(DCT)来实现类似的目标。这两个项目都没有取得重大进展，但是它们也显示了双方希望加强合作的意愿。

建立更紧密的联系纽带

最终改善双方关系的是 Ubuntu 社区中的文化转变。更多重点被放在向 Debian 作出贡献上，某些个体 Ubuntu 团队开始与上游 Debian 开发者在特定项目上进行合作。Ubuntu 开发者鼓励志愿者积极向 Debian 团队提交漏洞报告，以及直接参与 Debian 开发。在某些特定领域，诸如 Python 封装，双方也积极进行了协作。

Canonical 的回报 Debian 也在一定程度上上缓解了双方的紧张关系。 Launchpad 开发平台被 Ubuntu 广泛应用，但是它的闭源状态是一个突出的问题。今年年初这个问题也得以解决，Shuttleworth 宣布整个 Launchpad 组合将在今年完全开源。

尽管 Ubuntu 和 Debian 之间的分歧已经大大减少，但是它们之间的关系依然不够完美，要想让人们完全放弃 Ubuntu 收获大于付出的观点，Ubuntu 社区和 Canonical 还有很多工作需要做。

Ubuntu 因为承袭了 Debian 的强大功能才得以诞生，而且它还将继续依赖 Debian 社区的力量来实现自身的提高。同时，Ubuntu 的流行也 对 Debian 的知名度提高起了推动作用，通过协作性的技术工作，Ubuntu 开发者也已经开始逐渐对 Debian 的发展作出贡献。了解 Debian 和它的目标、价值和力量，将有助于明确 Ubuntu 的过去和未来。

Novell 裁减不足 100 名员工

北京时间 2 月 1 日早间消息，据国外媒体报道，Novell 公关主管伊恩·布鲁斯(Ian Bruce)周六称，该公司在全球范围内裁减了不足 100 名员工——不足员工总数的 3%。

Novell 并非唯一一家裁员的开源软件公司。Sun 已经裁减了 18%的员工，数家知名开源软件公司也“秘而不宣”地裁减了大量员工，以迅速扭亏为盈。Red Hat 尚未公布任何裁员计划。



分析师称，多年来，Novell 运营成本一直过高。该公司“会议泛滥”，许多员工的工作似乎就是参加各种会议。这次裁员将有助于 Novell 提高运营效率。

微软高层：GPL 协议机遇与挑战并存

微软平台策略部门高级经理 Sam Ramji 近日承认开源世界的规范之一，GNU/GPL 授权协议是一种重要的协议条款，不过企业认为与之相容是件充满挑战的事。就是在 8 年前，这家公司的 CEO 鲍尔默居然还宣称 Linux 是一种"癌症"，GPL 还要求过微软根据其规定把软件产品完全开源。不过现在，微软已经通过其合作伙伴启用了一部分基于 GPL 的技术，并且软化了自己的观点。



GPL 的基本精神是：只要有一小部分代码引用了基于 GPL 协议的其它代码，那么整段代码都需要开源，以便开源社区人士进行修改或完善。Ramji 比起鲍尔默，在看待此协议的时候要实用主义得多。他强调尽管 GPL 对于遵循这一协议的代码是重要的，不过使用此协议的代码仅占所有宣布开源的代码的 1/27。 然而，这一断言忽略了使用 GPL 协议的代码质量。"这等于在说"中国和印度只不过是全世界 190 多个国家和地区中的两个而已"这样的话，"自由软件基金会 的分析师 BradleyKuhn 说道。

Ramji 将 GPL 带来的挑战总结为，GPL 主要是一种专用的协议，不适合于多协议混杂的程序设计。而混杂的系统，交错的版权协议，多样的开发模型等，能让开源与闭源共同工作是很重要的。微软可以通过像 Novell 这样的合作伙伴使用 GPL 代码，但还不能直接向 GPL 计划贡献源代码。微软还曾作出承诺，除了微软已申请的专利之外，微软会承认自己的部分软件基于 GPL 协议。不过 Kuhn 认为，微软不愿意让自己和独立的软件开发者处在同等地位，不愿让个人或小团体掌握软件开发的核心控制权。因此微软在合作时更愿意选择本身有版权的项目。

微软毕竟还是为 GPL 做了一点贡献。他们曾经收购了一家制作基于 GPL 的 UNIX 桥接工具的公司 SoftwaySystems。Kuhn 指出，上世纪 90 年代末到本世纪初的一段时间，微软曾经顺从的分发(有时是帮助制作)基于 GPL 的软件。近年来，他们则更愿意跟随开源软件同步推出自己的非 GPL 方案。Kuhn 认为微软并不愿意让人们注意到微软现在已经拒绝贡献 GPL 协议下的软件这一事实。

对 Kuhn 的质疑，目前 Ramji 尚未回应。

PayPal 的开源极具成本效益

维萨公司的前任首席技术官表示，贝宝（PayPal）的升级道路“极具成本效益”。

2005 年 Scott Thompson 离开维萨公司（Visa）、担任贝宝公司（PayPal）的首席技术官职务时，这家互联网公司的数据中心让他大吃一惊。他回忆说：“他们居然在 Linux 上运行支付系统？！”Thompson 说：“我对支付系统和全球交易系统熟悉得很；可到了贝宝后，我愣是想不明白为什么要用 Linux。”他之前负责过 IBM 大型机和庞大的 Sun Solaris 系统，而贝宝对计算机系统采取的做法似乎很另类，对与钱打交道是其命根子的这家公司来说更是如此。贝宝运行着数千台基于 Linux、单一机架式的服务器，这些服务器上面运行着该公司的 Web 表现层、中间件和用户界面。

Thompson 表示，自己很快看到了开源软件和 Linux 技术在经济、运营和开发等方面的优势。他现在觉得还真离不开它们了。他说：“如果你购买许多大型机，就像我之前在其他公司遇到的那样，升级道路是一次就要砸下二三百万美金。为了进行扩展，只好购买大批的设备。而在贝宝这里，我们的升级道路是，只要把 10 台单价 1000 美金的不知名服务器添加到平台的中间层。我们就是以这种方式不断扩展。这极具成本效益。” 贝宝负责核心技术的副总裁 Matthew Mengerink 表示，这种模式还使得网站高度可靠。他帮助从头开始构建了这套架构。 Mengerink 说：“我们拥有数量众多的节点，个别节点出现问题也没什么关系，而不是拥有单一整体设备，或者是牢不可破的堡垒。”他说，使用专有操作系统来扩建拥有数千个故障点的系统不是个办法。他又说：“我们拥有的这种分布式、高度冗余的系统基于 Linux 和英特尔架构的成本模式。”

这种分布式模式还让这家公司可以在需要时，进行重大转变、分配资源。通用型、积木式的 Linux 服务器组成了该公司的 Web 层，很容易切换，以处理各项任务。比如每天凌晨 1 点，贝宝都会运行批处理以便调和支付。Thompson 表示，这种工作由数据中心的诸多中间层 Linux 服务器来共同处理；而在其他支付公司，这项工作通常在大型机或者大型对称多处理计算机上执行。 他说：“我们无须让网站停止运作。我们只要分配比较多的服务器资源来运行批处理；每天晚上只要三个小时就能处理完所有这些数据。”

在后端方面，这些成千上万个系统与区区几台大型 Sun Solaris 设备进行联系，这些 Solaris 设备运行的 Oracle 数据库负责保存所有客户数据。一套定制的数据库连接管理系统把来自贝宝网站基于 Linux 的 Web 层和中间件层的 Web 进程，与 Sun/Oracle 后端系统联系起来。 Mengerink 说：“进程进出处理的速度极快。所以，Web 层与数据库层之间有一个层来缓冲。而应用程序只管调用数

据库，它根本不用理会这个中间层。数据库那边看到的是稳定、老式、持久的连接，不会因暴风骤雨似的连接而瘫痪掉。”使用开源 Red Hat Linux 还带来了开发方面的几个优势。由于贝宝在生产系统上使用低成本的软硬件，所以在应用开发测试环境几乎可以复制整个活动网站（live site）的状态。这让贝宝的编程人员可以编写贝宝生产应用软件的新版本，然后可以把新版本放到活动网站上，基本上没有什么干扰。

Mengerink 说：“如果你的开发人员在与生产环境一模一样的环境上测试系统，到时出现异常情况的可能性要小得多，这至关重要。另外，开源软件还显然大大提高了测试工作具有的成本效益，因为不用支付在测试环境复制活动网站所需要的许可费。”这种模式还帮助贝宝的开发人员可以经常迅速开发出该网站几款主要应用软件的新版本，不过这有利也有弊。 Mengerink 说：“我们面临的老大难问题是决定如何使软件开发与活动网站相一致。开发人员比较激进。他们一直中意最新颖、最优秀的测试版，然后借鉴在某个大学网站上发现的某个内核补丁。但业务部门的人员要保守一点；他们觉得，最好就用最稳定、最成熟的软件版本。” 他表示，贝宝开发人员在构建 Linux 内核以及他们使用的其他开源代码时采取了这种做法，从而有助于提高整个系统的安全性。

贝宝数据中心中的 Linux 服务器运行 Red Hat 内核，而内核作了自定义的细微改动，从而为系统增添了额外的安全性。作为一个基本措施，过多的服务、程序包及其他软件都被摈弃了。他认为，Linux 与开源软件的组合让我们易于进行改动以便扩展，同时具有极高的安全性。

到目前为此，分布式 Linux、开源软件和开源代码的快速应用开发这对组合大获成功；这对组合肯定会继续收到良好成效。

提升 Qt 的应用 诺基亚即将增加授权选择

诺基亚 2009 年 1 月 14 日宣布，从预定于 2009 年 3 月发布的 Qt 4.5 版本起，其用于桌面和嵌入式平台的 Qt 跨平台用户界面（UI）及应用程序框架将在开源 LGPL 2.1 版授权下提供。此前，Qt 一直是在通用公共授权（GPL）下提供给开源社区的。



向 LGPL 的转移将为开源和商业开发人员提供比 GPL 更多的授权权限，从而为开发人员提高了灵活性。此外，Qt 源代码库将更加开放，鼓励更多来自桌面和嵌入式开发人员社区的贡献。随着这些变化，开发人员将能够积极推动 Qt 框架的演进。

Qt 4.5 同时也可在商业授权条款下使用，Qt 之前版本的授权则保持不变。而且，Qt 服务将扩展，确保所有 Qt 开发项目，无论选择何种授权，都能获得同等支持。

“更多领先企业对 Qt 的更广泛的使用将带来宝贵的反馈信息和更多的贡献，从而确保 Qt 始终是最佳的跨平台用户界面和应用程序框架。加速 Qt 开发将使得开发人员，包括诺基亚本身，能够创造更好的设备与应用程序，缩短推向市场的时间，保证为其解决方案创建更广泛的开发基础。”诺基亚 Qt Software 副总裁 Sebastian Nystrm 表示。

“诺基亚通过对 Qt 持续不断的支持，通过其对 Symbian 操作系统以及 S60 对 Symbian 基金会的贡献，和 通过 Maemo 平台的开源式开发，为开源社区做出了极其重要的贡献”，诺基亚终端部门执行副总裁 Kaiistm 说，“通过采用 LGPL，开放 Qt 源代码库，以及鼓励更多的贡献，Qt 用户在使用 Qt 开发时将获得更多的价值，同时也将会反过来鼓励对 Qt 的更广泛的推广。诺基亚同样可以在将改进后的 Qt 部署在 S60 Symbian 操作系统，Maemo 和 OVI 服务时获益，而无需重写源代码。”

“结合诺基亚独立于操作系统的应用程序框架 Qt 和飞思卡尔的可实施软件，为 OEM 和应用程序开发人员在挑选飞思卡尔芯片时提供了特有的自由度，从而允许开发人员为其应用程序开发和维护单一的代码库”。飞思卡尔解决方案及可实施技术副总 裁 Raja Tabet 说， “LGPL 模式是一个出色的和时效性的授权选择，这将加速结合了飞思卡尔和 Qt 的平台的推广与开发。”

“Qt 被广泛应用于 Kubuntu 和 KDE 应用程序中，Canonical 很高兴看到其在授权模式上的这一突破”， Ubuntu 项目创始人 Mark Shuttleworth 说，“Qt 新的授权条款将帮助我们为用户提供空前‘诱人’的应用程序。诺基亚对跨平台 Qt 库和 Linux 平台的一贯投入，是免费软件桌面和移动设备堆栈创新的主要动力。”

“我们欢迎诺基亚简化 Qt 授权的举措”，Linden 实验室平台与技术开发部副总裁 Joe Miller 说，“我们发现 Qt 是耐人寻味且极具创新的技术，无论授权方式如何，这个新的授权方式已经使得我们在追求将 QtWebkit 集成到 Second Life 时所作的决策变得更为简单。”

“Qt 在 LGPL 条款下的使用，让运用基于 Qt 应用程序顶端的 KDE 组件创建应用程序的授权合理化”， KDE e.V. 董事会成员 Sebastian Kügler 说，“这一更多权限的授权为 Qt 和 KDE 技术的推广再次降低了门槛。KDE 团队欢迎开放开发进程，并期待以此进一步促进 KDE 和 Qt Software 部门的协作。”

社区扫描

Ubuntu Mobile 考虑使用 Qt



Canonical 旗下除了 Ubuntu 外，还有众多衍生版本，其中一种是专门针对上网本和移动互联网设备的版本 Ubuntu Mobile。目前它使用的是 GNOME Mobile，自 Qt 4.5 宣布将采用 LGPL 许可证之后，Canonical 的 David Mandala 表示，他们正在考虑 Qt，寻找比 GNOME Mobile 更好的框架。

Gentoo Portage 包管理系统已支持 Git

Portage，Gentoo Linux 的包管理系统，其最新版本 2.2_rc20 已经提供了针对 Git 版本控制系统的支持。目前，完全独立的 Funtoo Portage 树即是由 Git 所管理。相信这将使各 Gentoo 开发者进行更好的协作。

更多信息可见于 Gentoo Linux 创始人 Daniel Robbins 的 Blog。

Python 3 的演变

O'Reilly 采访了 Python 语言作者 Guido van Rossum，讨论了 Python 3 以及未来的发展。在最后，他表示：

我想重申一点，决定是使用 3.0 还是 2.6 都是个人的选择。你不会因采取保守立场而有被抛在后面的风险。3.0 和 2.6 都被同一个核心的 Python 开发 者小组所支持。此时我们也不特别强调 Python 3 的重要性和品质。如果不是受到外在要求，或者第三方软件尚未移植到 3.0，或工作在一个其他人都是其他版本的环境中，你不会受到阻碍。如果你是第一次学习 Python，3.0 是一个容易上手的语言。一些困扰初学者的东西已经被去除。如果你学习了 3.0 再去看其它版本，会很容易了解 2.6 和 3.0 的区别。 如果你学习 Python 2.6，你可能会为过时的语法所阻挠。

热点回顾:Linux 2.6.28 五大特性

Linus Torvalds 将 Linux 2.6.28 作为圣诞礼物送给所有 Linux 用户，现在新年已过，是时候回顾这个去年最重要的内核版本了。

Computerworld.com 的一位博客列出了他认为的 Linux 2.6.28 五大特性：

1.Ext4：新的文件系统改进了硬盘储存，支持更大的文件，更快的 I/O，更好的日志，不需要整理碎片。使用 MySQL 5.0 的非正规测试显示，在 400GB 数据库中写入速度提高了 30%。

2.GEM 显存管理器：以廉价的 Intel 915 芯片组为例，在 GEM 的帮助下它的速度提升了 50%。

3.磁盘防震保护：如果探测到磁盘快速的移动(比如笔记本掉到地板上)，它会让硬盘的读/写磁头降低速度。

4.分阶段驱动（Staging Drivers）：Linux 对硬件的支持虽然不错但称不上完美，如果你使用一个新硬件就可能会遇到驱动问题，分阶段驱动可以部分的解决这一问题。

5.网络改进：2.6.28 kernel 支持 UWB (Ultra Wide Band)、Wireless USB、UWB-IP，和诺基亚的移动电话 Phonet Network 协议。

Perl 迁移至 Git 版本控制系统

Perl 基金会宣布他们弃用 Perforce，换用 Git 版本控制系统。使用 Git 后，Perl 语言的开发团队将能利用 Git 的离线和分布式版本支持功能（关于版本控制，阮一峰的网络日志有一篇入门教程）。

Git 是 Linus Torvalds 开发的开源分布式版本控制/软件配置管理软件，能简化代码递交和管理开销。从 Perforce 转移到 git 将要花费一年时间， git 仓库将包含历史意义快照和模块补丁集（patch set），现在开发者就可以从 perl.org 网站下载一个当前版本的 Perl Git 仓库。

BSD 发行版发布：FreeBSD 7.1

最新版本的 FreeBSD 7.1 已经发布！欢迎大家在 CU 下载频道下载。

FreeBSD 是一份 UNIX 操作系统，它面向 i386、IA-64、PC-98、Alpha/AXP 及 UltraSPARC 平台。它基于加州伯克利大学 的 4.4 BSD-Lite 发布，并带有一些 4.4 BSD-Lite2 增强。它还非直接地基于 WilliamJolitz 的 port，这源于加州伯克利大学 i386 化的 “Net/2” 也即 “386BSD” ，尽管 386BSD 中只有非常少的代码遗留下来。FreeBSD 被遍布全世界的公司、Internet 服务提供商、研究人员、计算机专家、学生，以及家庭用户用于他们的工作、教学和娱乐之中。

FreeBSD 7.1:从 Sun 获取一些帮助

BSD 的最新版本增加了新的 Sun 开发的功能,但这并不是唯一的技术转移。

开源 FreeBSD 操作系统在经历了将近一年时间后，终于带着其首次主要更新与大家见面了。FreeBSD 的 7.1 在其前身 FreeBSD 7.0 基础上进行了许多改进，其中包括 Sun 微系统公司开发的 Dtrace 技术以及新的启动选项和可扩展性的改进。 FreeBSD 7.1 还展示了开源是如何跨越不同公司以及不同的操作系统。FreeBSD 是最早的开源操作系统项目之一，其前身是美国加利福尼亚大学伯

克利分校的开源 BSD 组。

FreeBSD 的核心成员 Robert Watson 在 InternetNews.com 上说：“DTrace 最初是由 Sun 公司开发的一个成熟和令人信服的技术，用于性能监测，最初作为 OpenSolaris 的一部分而开源发布，虽然以前我们有许多用于多种具体分析的工具，但 DTrace 是一个极好的管理和跟踪数据的通用框架,还让我们 能够更方便地添加新类型的跟踪。”

Watson 说,如果没有 Sun 的 DTrace 对开源世界的贡献，可能就没办法集成 DTrace 到 FreeBSD 上。John Birrell 一直在与 Sun 紧密联系，跟进这件事。Sun 微系统公司的高级工程师 Bryan Cantrill 在 InternetNews.com 上说，除了 Birrell，还有一些 FreeBSD 的人参加了去年 Sun 公司的 DTrace 会议。

DTrace 并不是 FreeBSD 中使用的唯一的 Sun 开发的技术。FreeBSD 7.0 版本就引入了对 Sun 的 ZFS 文件系统的支持。另外，Sun 和 FreeBSD 之间的技术转移不止一种方式。Watson 认为，OpenSolaris 的内核采用 FreeBSD 无线网络架构，支持 CIFS 文件系统，这让 OpenSolaris 获益良多。

但 sun 否认 CIFS 栈来自 FreeBSD。Sun 公司发言人指出,它来自于一个几年前 Sun 收购的公司 Procom。这位发言人说，很多 OpenSolari 的 WiFi 驱动程序和核心 WiFi 基础架构都来自 FreeBSD。



更多 FreeBSD 7.1 特性

除了 DTrace 的集成,FreeBSD 7.1 还列举了 USB 启动作为一个新特性。FreeBSD 的贡献者兼 Absolute FreeBSD 的作者 Michael Lucas 认为，FreeBSD 很多年前就可以从 USB 启动了。

Lucas 在 InternetNews.com 说， “尽管某些特定硬件不喜欢通过 USB 接口启动 FreeBSD，但对于一个新特性，我们很难说它适用于 90%的硬件。USB 启动现在已经更加可靠了。” Lucas 还认为 FreeBSD 7.1 的 UDP 网络协议栈也有了改进。

Lucas 说，“过去几年，我们对于多处理机硬件上的网络协议栈的可扩展性做出了巨大的改进，但大多数的测试都是在典型的基于 TCP 的网络负载上进行 的 ——Web，电子邮件等，网络组针对 ISC 已做了大量的工作，包括改善根域名服务器操作的 UDP 性能。这是一个不太明显的改变，但它

可以让一些互联网上最重要的基础设施处理更多的负荷。”

FreeBSD 8.0

虽然 FreeBSD 7.1 刚刚发布，但开发人员正在努力工作于下一个主要版本 FreeBSD 8.0 ，其中将包括网络虚拟化的改进。



Watson 说，“我们非常期待 FreeBSD 8.0，其将于年底发布，它将支持虚拟网络协议栈，这将让 FreeBSD 拥有自己的路由，防火墙，虚拟专用网等。这对于我们的 ISP 用户、设备厂商、研究 界等来说都是令人兴奋的消息。另一个同样会令人兴奋的功能是支持 802.11 虚拟接入点，它将允许许多不同的 802.11 SSIDs 使用同一个频段，这对使用 FreeBSD 建立商业接入点产品的公司来说,是很重要的。”

Mozilla 向 Ogg 项目捐赠 10 万美元

Mozilla 加入了维基媒体基金会资助 Ogg 开发的计划，向该项目捐赠了 10 万美元。

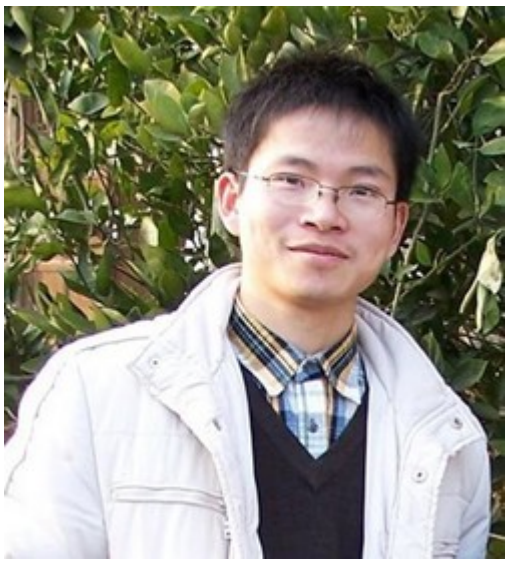
Mozilla 的理由很简单：音频和视频开放标准的重要性不言而喻，它们能被任何人为任何目的使用，而无需缴纳版税，并且还能被开源社区检查和改进。今日 Web 上的音频和视频已为私有技术所统治，闭源播放器包含了大量专利的解码器。因此维基媒体和 Mozilla 决定主动帮助去改进开放的、基于标准的视频技 术的质量。维基媒体基金会的所有富媒体都将使用 Ogg 格式，从而成为互联网上最大的开放媒体格式采用者之一。Vorbis 是比 MP3 更好的标准，但 Theora 却被普遍认为远逊于其它视频解码器，开源社区还有许多优秀的解码器如 Dirac、FLAC 或 Speex，它们却常常被忽视。

专家专栏

Conke.hu：做面向 Linux 爱好者的开源项目

新一期的《专家专栏》又和读者朋友们见面了，今天我们有幸请到的是 ChinaUnix 开源项目板块中 g-bios 和 Maxwit 创始人 Conke.hu，就这两个开源项目的问题做一个访谈，也好让 Linux 和嵌入式技术爱好者们更好的了解这两个开源项目。

Conke 在 ARM 及 X86 体系结构与编程，及嵌入式系统上 bootloader/BSP 开发，Linux 内核开发和调试，包括 Linux-2.4/26 中断、memorymanagement、内核进程/线程调度、块设备/文件系统、I/O 设备驱动上，有着丰富的经验。Conke 曾为 AMD 第一款桌面主板芯片组(A690G SB600)Linux 驱动项目的官方首席工程师，其部分开源代码发布在 linuxkernel.org 及 linux2.6.18~2.6.21 中。



ChinaUnix：能给大家简单的介绍一下 g-bios 和 MaxWit Linux 这两个开源项目的情况么， 为什么这么取名?

Conke.hu：我们 MaxWit 开放实验室发起了两个开源项目：g-bios 和 MaxWit Linux。MaxWit Linux 是一个基于 Linux 的嵌入式操作系统，类似 Google 的 Android 或 Intel 的 Moblin。具体情况在上一期《开源时代》中已有较详细介绍，我这里简要介绍下 g-bios。

g-bios 是一个 Bootloader，或者说是一个嵌入式系统的 BIOS，类似于 u-boot（另一个有名的 Bootloader，由德国人发起)。g-bios 的作用相当于 PC 机的 BIOS + Bootloader。g-bios 不但借鉴了几乎所有主流 Bootloader/BIOS 的优点，而且加入不少独创的特性，包括：

a) 自动检测有待烧录的 image 文件类型，并智能自动烧录。

b) 支持多种文件系统，包括 YAFFS2、JFFS2、CRAMFS、NFS 等。

c) 支持两种用户界面：GUI（类似传统 PC BIOS）和命令行模式（面向嵌入式系统）。

d) 命令行自动补全（Tab 键）及历史记录（上、下键）支持。

e) Flash(MTD)分区支持，帮助 Linux、Android 内核识别分区。

f) 自动设置 Linux 内核启动参数（Linux kernel command line），极大地降低了参数设置的复杂度并减少了启动出错的概率。当然，同时也支持手动设置，以满足特殊要求。

g) 常用命令具有记忆功能。如 boot 命令，它能记住用户输入的参数，以后只需简单输入 boot 即可。

h) 引入全新的架构及 NB（Never Burn Down，烧不死）技术。开发人员可在没有仿真器的情况下轻松完成整个 g-bios 的开发。

i) 优秀的子系统设计，包括：中断、网络、Flash、USB 子系统，等等。

j) 集成类似 PC 机版本的 Video BIOS。

k) 支持基于龙芯的 PC 机或嵌入式系统。

l) 完美支持 Google Android 操作系统，简化 Android 的系统移植过程。

m) 提供图形化配置界面 gconfig，不但让新手很容易上手，而且使 g-bios 的移植和开发过程变得更简单。

至于这两个项目的取名，MaxWit Linux 这名字源于我们的 MaxWit 开放实验室，中心词 Linux 表示这是一款基于 Linux 的嵌入式操作系统。g-bios 中的 g 表示 GPL 和

Generic（通用的），中心词 bios 表示她是一款 BIOS/BSP/Bootloader 类软件，即“基于 GPL 协议的、通用的 Bootloader。

ChinaUnix：能简单介绍一下你的开源开发的经历么？目前这个这两个项目有几个开发人 员，大概的分工情况是如何的?

Conke.hu：还在读小本时我就对 Linux 非常感兴趣，浙大有较浓的 Linux 学习氛围，同时又结识了几个志同道合的校友，从此开始研究以 Linux 为代表的开源项目。在之后的几年工作期间我一直专注在 Linux 及相关开源项目的研发上，在 Intel 和 AMD 工作期间多次向 Linux 内核及其他开源组织贡献代码。最近成立了 MaxWit 开放实验室，专门从事开源软件的研发。

目前 g-bios 和 MaxWit Linux 的研发团队是同一个，核心成员共 6 人：Linke Wang、Tiger Yu、Fleya Hou、Pony Ma、Homer Xing 及本人。Linke 主要负责驱动开发及应用软件移植，Tiger 负责图形引擎的设计和应用软件开发，Fleya 负责驱动开发及不同硬件平台的移植，Pony 主要负责项目的宣传及部分编码，Homer 负责项目的编译脚本及上层应用软件移植，至于本人，是个“打杂的”，其他成员负责的某个模块需要帮忙我就跑去那里。项目整体架构的设计由我和大家一起讨论决定。

ChinaUnix：目前这些项目采用的授权协议是什么？为什么?

Conke.hu：如前所述，这两个开源项目均采用 GPLv2 协议。采用 GPL 协议的原因很简单，希望有更多的人参与到该项目中来，同时也方便其他企业或个人运用到别的 GPL 开源项目中去。

ChinaUnix：目前这两个项目和国际上类似的项目相比，有什么不同呢?

Conke.hu：国际上与 g-bios 同类的项目中最有名的要属 u-boot。u-boot 最大的优点在于支持的硬件平台比较多，缺点是大部分关键子系统架构设计不完善或者尚未实现，如中断、网络和 USB 等，

而且编码不规范，代码质量不高，还有，不支持 PC 机。

g-bios 优点在于整体架构和各子系统的设计都很完善，包括 Shell、中断支持、网络、Flash 等。缺点是目前支持的硬件平台不够多。

与 MaxWit Linux 类似的项目有很多，如 Intel 的 Moblin 和 Google 的 Android。Moblin 的设计基于传统 PC Linux 架构，并结合了某些嵌入式 Linux 的优点，如内存使用的优化、GUI 和电源管理的改进等。Moblin 专注在 X86 架构上，为推广 Atom 处理器和 MID 设备而服务。Android 采用与传统 Linux 截然不同的架构，其体系结构分四层，从下到上依次是 Linux 内核、基于 C/C++的程序库和 Java 虚拟机、应用程序框架、应用程序。其中最底层的内核和基于 C/C++的程序库与其他嵌入式 Linux 没有本质区别，但中间层和上层 Google 则另起炉灶，用 Java 全新设计所有的组件，包括 Windows Manager，这一点与 Sun 的 Java OS 类似。Android 最初专为智能手机而设计，以后还会向其他领域扩展。

MaxWit Linux 与其他项目最大不同之处在于面向的对象，MaxWit Linux 专为嵌入式爱好者学习和研究嵌入式软件开发而设计。MaxWit Linux 的设计参考了现有主流嵌入式操作系统的优点，并试图逐一剖析各个核心技术的设计与实现，借助文档和 BBS 和大家分享，而且集成了自动编译脚本，即使是没有基础的初学者也可以毫不费力地入门和研究。

ChinaUnix：在国内和在国外做开源项目有什么区别呢?

Conke.hu：首先，从整个行业看，国外开源社区发展得比较成熟，开源项目数量多，质量高，如 Linux kernel、glibc、Xorg、GTK+等等；而国内有质量的开源软件非常少，skyeye 后来没什么人继续开发和维护，MiniGUI 纯商业动作了，而且也不能算真正的开源项目。

第二，从发起人和维护者的角度比较。国外开源项目的维护者更富有热情，而且长时间坚持着，而国内的发起人很少有人能长久保持这种热情。

第三，从支持者和参与者角度比较。国外参与者多，一个开源项目中源码大部分来自普通参与者而不是项目发起人或维护者；国内开源项目参与者少，且大多停留在使用而非开发的层面上。

第四，国外开源项目的背后往往有商业公司的影子，这不但激励了开发者，而且也使项目本身在企业的实际应用中得以快速完善和发展；而国内的开源项目大多与企业应用脱节，也缺乏与企业的合作。

ChinaUnix：目前这两个项目打算朝什么方向，怎么发展?

Conke.hu：g-bios 定位在企业级 Bootloader，而不只停留在一些技术狂热爱好者的研究层面，而且同时支持嵌入式系统和 PC 机！MaxWit Linux 面向嵌入式 Linux 自由爱好者和企业中的 Linux 研发人员，主要用于学习和研究。

除了使用传统的开源项目开发方式（项目主页 + Mail List 和 BBS）之外，我们还为 g-bios 和 MaxWit Linux 创造了两个非常有利的发展因素：一是在上海和北京专门设立了完全免费的实体研讨空间——MaxWit 开放实验室，并配备了比较齐全的软硬件开发环境，可供所有爱好者和项目参与者使用；二是和多家企业建立了合作关系，共同开发 g-bios 和 MaxWit Linux。

ChinaUnix：能给我们客观的谈一下，目前项目还有那些需要重点发展的地方?

Conke.hu：g-bios 各子系统已经比较完善，目前最需要做的是支持更多的硬件平台；还有为了支持龙芯等 PC 级系统，还需要加入 GUI 界面和 USB 驱动。

MaxWit Linux 重点开发的图形优化和多媒体功能的增强。无论是嵌入式还是 PC Linux 都没有像 MS DirectX 那完善的多媒体开发库，目前我们正在改进嵌入式 Linux 的图形驱动架构，加入对 OpenGL 和 SDL 的硬件加速支持，并试图进行桌面级整合。

ChinaUnix：现在有无基于这两个开源项目的商业化的支持和解决方案？或者将来是否有这方面的打算？打算从哪些方面入手？

Conke.hu：这两个项目的开发者一直按照“商业级”的要求在做，即不低于企业实际使用的其他同类方案的技术水平，但并没有“商业化”的打算，而将其永远定位在 GPL 开源免费项目。

ChinaUnix：除了这两个开源项目外，还对那些开源项目有热情？

Conke.hu：Linux kernel、libc 库（glibc 和 uClibc）、Xorg、DirectFB、SDL。

ChinaUnix：对从事开源技术的网友想说些什么？如果他们也想做一个开源项目，或者基于开源项目的初创公司，他们应该注意什么？

Conke.hu：在开源社区里我看到过很多优秀人才，其中有不少是来自我们中国，但国际上众多优秀的开源项目中出自国人之手的几乎没有。比方说，开源社区中每 100 个中有 2 个是中国人，但 100 个开源项目中却没有一个是从中国的开源力量中诞生，这是一个值得大家思考的现象。事实上，国内尝试发起开源项目的人数并不少，但往往是单枪匹马或支持者少，以至进展缓慢，甚至放弃。有志于做开源项目的网友需要做到以下几点：

1) 要有规范的项目管理。
2) 做好项目的宣传，让更多的人了解并参与项目。
3) 要有好的交流平台，如 BBS 或 Mail List 等，可以更好地合作开发。
4) 与其他企业合作开发。

行业观察

企业如何监管开源软件使用情况？

不管企业是否了解开放源代码技术，几乎所有企业都安装了开放源代码技术。在成千上万的开源项目基于派生于数不清的受制于各种许可证的源代码的软件的情况下，必须采取治理措施来减少法律和运营风险。其目标是部署政策、系统和流程来确保管理使用开源软件的合理的标准。

公司可以部署各种治理解决方案，这些解决方案使他们可以记录开源的使用情况，实施开源政策，自动完成批准流程，跟踪和审计开源部署以及确保与开源许可证的遵从性。

开源治理的关键组成部分是了解企业使用什么开源软件，在何处使用。除了帮助你确定法律风险外，掌握“使用什么”和“在何处使用”在系统停机、发现安全漏洞或发生法律诉讼时至关重要。

令人吃惊的是，大多数企业不知道它们使用多少开源代码。在一些开源技术被各种规模的公司广泛使用的同时，另一些开源软件包由开发人员从网上下载，从而绕过了标准的采购流程和控制。此外，一些工具由于被捆绑在开发人员需要的其它软件包之中，因此常常在不知不觉的情况下被部署。

事实上，开源程序可能包括几十种其它开源组件，从而使评估使用什么开源软件和涉及哪些许可证变得非常困难。建立开源库存目录基线是治理的关键步骤。此举将帮助你确定减少法律风险和计划使用、支持和更新开源组件所需要的行动。

开源软件的许可证难题

虽然开源代码可免费下载，但每个开源项目都附带着管理使用条件的开源许可证。事实上，目前有 50 多种获得 OSI 批准的开源许可证、这些许可证的成千上万种 变种以及大量的没有得到批准的一次性开源许可证。尽管很多开源许可方式的讨论围绕着 GPL 许可证展开，但其它的许可证在企业使用的开源软件中更常见。



与使用商用软件不同，你必须研究哪种许可证适用于下载的开源软件。在许多情况下，由于捆绑的代码而可能涉及许多许可证，并且这些许可证常常没有被明确地罗列出来。因此，发现所有适用的许可证需要谨慎对待。此外，一些许可证可能要求承担相互冲突的义务，即使它们应用于某个应用的组件。

一旦你了解你拥有什么开源工具后，你必须定义和实施管理使用它们的政策。这些政策应当定义参数和规则。这些参数和规则包括对证书的要求或开源软件的支持、 什么许可证是可接受的、开源软件获得批准的流程和标准以及与开源社区打交道的指导方针。你的机构将根据这些参数和规则选择、使用和管理开源软件。例如，一项政策可以定义预先得到批准的开源工具的白名单、禁用的开源工具的黑名单以及哪些灰名单开源软件包可以使用的条件。

许可证审批的管理的范围从宣布使用什么开源软件的简单电子邮件流程到更复杂的流程。重要

的是记录使用什么开源软件、何处使用和什么得到了批准。随着这些批准流程被实施和自动化，它们将成为标准开发实践的组成部分。

你最后需要的一样东西是完整的开源工具的审计跟踪。如果没有安装了什么软件的记录，那么当出现法律或生产问题，搜寻这种记录将变得非常困难和代价高昂。

首先，企业必须跟踪经过指定的流程（包括从批准的开源库中下载的软件和批准使用的历史）开源软件。但是，即使有了合适的政策和流程，一些未经批准的开源软件可能会通过缝隙溜进企业。因此，企业还必须通过定义的检查和审计来支持这些努力，来确保所有人都遵守规则。审计使企业可以将开源软件使用情况与使用政策 和批准使用的软件进行比较，寻找不一致的现象。

此外，正如使用专有软件一样，你需要确保你支持开源软件、控制使用的开源版本和管理更新过程。许多企业说它们拥有一种开源组件的各种版本，它们常常不知道哪种版本正在何处使用。最终结果会影响到正常运行时间、效率和风险。

跟踪和审计流程应当使企业可以看到开源软件使用情况的多种视图——按项目、许可证、用户、生命周期状态和服务器。多种跟踪机制——请求表格、系统扫描、手工输入、构建工具、下载——是执行政策和管理遵从性所不可缺少的。

所有这些元素一起为企业提供实施开源软件治理和减少风险的宝贵的、可审计的信息。通过部署跟踪和审计开源工具的流程和工具，企业可以在减少风险的同时，利用开源软件带来的巨大的费用节省。

Linux 真正价值不仅在于产品本身

看一看四个“大的”Linux 公司和他们在操作系统市场上的地位，你会觉得很有趣，尤其是比较他们雇佣员工的数目、各自公司在市场上的影响（尽管与 Windows 比起来相对小很多）。



Canonical 是 Ubuntu 和 Ubuntu 类产品（Kubuntu, Xubuntu, Edubuntu, Ubuntu Server 以及许多其他的)）的资助者，公司在全世界拥有 200 名左右的雇佣员工。Red Hat 雇佣了大概 2500，Novell 雇佣了 4500，Mandriva 的这个数字大概是 130。

所有的这些加起来不过是 7330。当然，这些公司还有非常可靠的来自于社区的开发者的支持，但我所比较的只是给予薪酬的员工的数目。

微软雇佣员工的数目大概是 90,000，比所有四个“大的”Linux 公司的总和多 12.28 倍。单从这个数字上就能推断出微软应当有更高的市场份额，而且微软确实在操作系统的份额上雄踞霸主地位：通常这个数字被认为是 85-90%。

当 linux 不断发展试图争夺更多市场占有率的时候，当微软不得不积极应对来自于 Linux 阵营的“威胁”的时候，个人和商业消费者是受益者。我们可以从他们的竞争中获益到更多有用的功能、低的价格、更多的支持以及更好的软件。

例如，Beryl 和 Compiz 项目对 WinVista 上的微软 3D Aero 界面产生了怎样的影响？Mozilla Firefox 对 Internet Explorer 7 产生了怎样的影响？Linux 病毒、木马和间谍软件率低对 Windows 的安全模式又产生了怎样的影响？这些产品因此而越来越棒，而且还在继续完善中。

Linux 的真正价值不仅在与它能提供的软件、产品和服务等，而在于它能够驱动微软前进的脚步，这就是为什么 Linux 应当受到支持而不是排挤。所以我个人想看到 Linux 变得更好（实际上我也在社区做很多事情），这样不但 Linux 会更好，Windows 也会因此而更好。

微软有一个任务：为它的股东赚取尽可能多的钱。想象一下，如果没有 Apple 和 Linux 的竞争，微软会变得怎样？他不再有动力前进，是否我们仍停留在使用 MS-DOS 的阶段?

感谢微软与 Linux 及其他操作系统之间的良性竞争，以及由此所带给我们的更多的选择!

关于国产软件、本国软件与开源软件的概念纷争

作者：胡才勇

近来，还是有些人对开源软件是不是国产软件、国产软件的定义等问题产生了兴趣，并在我们公司的论坛里有热闹的争论。

直至今日，据我所知，并没有所谓的国产软件的官方的、正式的定义，而且，曾经有的草稿（讨论稿或草案）所采用也是按照国际惯例的“本国”软件而非“国产”软件。我所曾参与讨论过的一个草稿，曾把开源软件和本国软件之间的关系用一句话来定义：（在享受一些政策支持的情况下）开源软件等同于本国软件。但就是这样的定义，最终也没有最后的官方版本。

之所以采用这样的论述，很坦白地说，在现在开放合作的国际形势下，简单的讨论国产产品或者本国产品，虽然大家可以很轻松的根据品牌来区分本国产品或者非本国产品，但如果要来严格定义“本国产品”的内涵或外延，在文字上颇费周折，并且也有一些简单的法则很容易被人钻空子。比如微软中国现在也是中国软件行业协会的成员，那微软的 Windows 操作系统和 Office 2007 等是否也是国产软件呢？那显然不是。此外如大家公认的可口可乐、麦当劳、肯德基、IBM、SUN、家乐福等国际知名品牌，他们也均需要按照中国的法律或“市场规则”，加入一些中国的社会团体或者行业协会，但我们当然不会据此认为他们是国产的。

我很支持 07 年年底出台的《自主创新产品政府首购和订购管理办法》，而在如何理解“自主创新产品”这个概念时科技部万钢部长有一个非常精辟，我认为也是非常权威的说法：**“在经济全球化和技术日趋复杂化、交叉化趋势的共同作用下，在高技术产业领域，跨国技术联盟日趋增多、合作创新正在成为主要的创新方式。‘自主创新’应当强 调的是创新者对技术创新和产品开发的‘主导权’，而不是技术本身的来源。在‘合作创新’的技术联盟中，获取创新主要收益的也只能是掌握主导权的一方。”（摘自万钢 2007 年 8 月 28 日的讲话稿：《利用全球资源推动自主创新》）**

在软件业，在有一个世界性的协作创新平台——开源软件的情况下，如果我们还来“搞狭隘民

族主义，通过闭门造车提高国家自主创新能力是不可取的 ”（引号内文字摘自万钢的上文同一讲话稿）。所以，在我眼里，如何基于开源软件来形成国产基础软件的自主创新的主导权才是我们应该考虑的。而基于开源软件 的自主创新产品如我们的 RedOffice，自然是本国产品，自然是国产软件。相对应的红旗 Linux、共创 Linux、中标 Linux 等都是国产软件。

至于许多人所抱怨的国内开源企业不开源的问题，我认为不能一概而论，现在应该是越来越多的中国开源企业切实的加入了开源社区，并且切实地在作贡献，别的企业我不清楚，但我们红旗 2000 这些年的贡献却是有目共睹的。

我们红旗 2000 公司早已突破单纯获取 OpenOffice.org 社区代码的模式，而进入社区决策层，融入社区工作，并在某些领域引导社区发展。红 旗 2000 公司也是目前对 OpenOffice.org 开源社区真正贡献的企业，已成为社区世界范围内第二大技术力量(SUN 公司是最大的技术力量)。

我们公司技术领头人金友兵博士是 OpenOffice.org 社区高级顾问团成员，成修治先生是社区工程师领导委员会成员，公司近 10 名工程师成为社 区专业的 Developer(社区根据自身的评价体系从低到高认定四类职位：Observer、Contributor、Developer、 Leader，目前全世界获得 Developer 资格的不足百人)，同时社区的性能优化项目负责人(Leader)和中文本地化项目负责人(Leader)均由我公司部门总监担任。



截止 08 年 9 月，我们已向社区贡献了 187 个任务代码，负责和主导了社区近 10 个重大功能开发，修改了数百个 Bug，完成了大量本地化工作。现以每月 20 个左右任务代码贡献的速度递增(附件是社区网站上公布的我公司的贡献列表)。

基于我公司对开源社区的贡献和影响力，我们以绝对优势赢得了 2008 年 OpenOffice.org 国际开源年会的主办权，并于 2008 年 11 月成功 举办此国际盛会，这是该大会第一次在亚洲国家举办。为此，我们投入的经费超过了 100 万。该次大会获得了社区很高的评价，社区负责人 Louis 直接采用了 “无与伦比”来形容。除此之外，我们还在 2006 年委托国家软件质量检测中心，共同测试了 OpenOffice.org 的发布版本，并把测试结果直接返回社区。因此，无论是在源代码贡献和直接的资金贡献上，我们都已经投入了很多，也得到了社区充分的认可。

总是有人对国产软件的发展和未来高度存疑，总是有人害怕国内开源企业不开源。有道是：爱之深，责之切。但非要说采用开源软件就不能叫国产软件，那是不理解万钢部长所说的“自主创新”应当强调的是创新者对技术创新和产品开发的“主导权”，而不是技术本身的来源。放着好好的开源不用，非得自己一切重头来？基于开源软件，我们获取了自主创新的主导权，非得把这些企业

排除在国产软件、本国软件之外，是没有道理的。

如今，基于开源软件发展国产基础软件已成为了一个发展方向，所有关心国产软件发展的人都应该认识到，开源软件已成为了软件发展的一个潮流，把我们自外于开源软件来搞自主创新是不科学的。

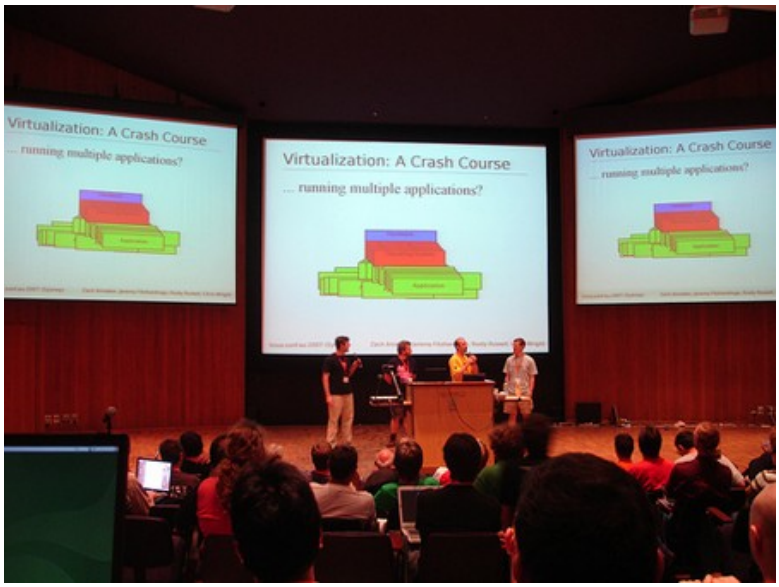
那些迫切希望国内企业能在开源社区里发挥作用的人，应该把眼光放长远一点，同时也一定要注意到已经有越来越多的国内企业开始在社区里有所作为了，切勿用五六年前的眼光来看待国产软件。至少，我们红旗 2000 的开源实践已表明，只要你真正投入，你就能被社区所接受，并逐步形成你的影响力。

全面普及企业级 Linux：仍重而道远

在过去的几年里，将商业应用从高端的基于 Unix 的系统比如 Sparc 或者 Solaris，迁移到商业 x86/Linux 平台已经成为了一种非常流行的概念，但不是每个人都同意在企业中全面普及 Linux 是最好的解决方案---至少目前还不成熟。

万豪国际连锁酒店公司企业运营副总裁 Dan Blanchard 对待 Linux 系统的态度非常认真。 他表示，自己所属的公司正在从高端的 HP-UX 系统向 IBM AIX 上进行迁移，这个过程是不可避免的。“我们正在实施迁移并计划继续部署 Linux 系统，不过完全过渡到 Linux 还需要几年的时间，那时技术将更完善，也可能出现其他机会。”

分析机构 Ideas 公司的资深分析师 Tony Iams 表示他曾多次听到企业 IT 主管表达类似的观点。他说：“我们公司有一个长期目标，那就是将所有 Unix 系统都整合到 Linux 上”， Ideas 公司再三声明他们的目标是通过行业标准技术，而这通常意味着在 x86 硬件上运行 Linux 操作系统。



但是，高通公司（Qualcomm）首席信息官 Norm Fjeldheim 决定放弃从 Solaris 到 Linux 的迁移。Qualcomm 确实在 Linux 平台上运行某些应用，但 Fjeldheim 的 IT 团队 认为：将企业级 Solaris 系统迁移到 Linux 平台上是一桩没把握的买卖，他表示“我们不会从 Sun 迁到 Linux 。因为我们还无法从经济角度找到这么做的理由”。

虽然乍一看从 Sun 迁到 Linux 会使 Qualcomm 省下硬件和操作系统费用，不过费用对比是基于供应商提供的零售价格的。“我们并不是以零售方式进行付款，当我们计算出采用 Sun 方案 的折扣

时，这种价格优势就消失了，因为 Linux 的速度非常快。” Fjeldheim 说。

但这并不是唯一问题。他的团队对于目前市面上流行的 Linux 管理工具不是非常满意。大约 18 个月以前， Qualcomm 的 IT 人员也评估过据说可以使管理员工作变得更轻松的 Linux 管理工具，但结果是“在 Linux 环境下真的没有 Unix 系统那么轻松 ”。而且，Fjeldheim 还补充说这将带来更多地管理费用。

Qualcomm 公司 IT 主管 Matthew Clark 作为审查小组的成员之一也参加了该 Linux 项目的评估。Qualcomm 目前管理员和用户的比例是 500：1 (尽管他计划将该比率将为 450：1)。“使用 Linux 环境，管理员与用户比为 150：1 到 175：1 之间。我们将不得不为目前每个 Unix 管理员额外再雇三个管理员。” Clark 说。

Iams 听到这种评价并不觉得惊讶，“优化系统---这是 Sun 的一个传统优势”。Clark 承认，自 Qualcomm 最后一次评估其 Linux 项目之后，管理工具已有所改善，不过他仍然认为 Linux 操作系统更为昂贵。“如果我们从今天起就使用新工具，结果可能导致我们每个人旁边都有两个管理员”。Clark 说。虽然这一次 Qualcomm 并没有采用 Linux 作为 Solaris 的替代品，Clark 仍表示 Linux 的总体性能给他留下了深刻的印象，并表示将继续考虑在某些环境下采用 Linux，“我们认识到，某些应用在 Linux 环境下运行得非常好”Clark 说。Blanchard 也认为，并不是所有应用程序都适合运行在 Linux 系统中。在某些情况下，他曾看到过万豪的 Linux 迁移并决定不实施此项目。

但总体而言，目前万豪迁移到 Linux 平台的应用往往是高端的 Unix 系统。Blanchard 认为取代这些 Unix 系统的 Linux 技术和管理工具已经足以满足万豪酒店的需求。“十年前我们就开始讨论企业级 Linux 系统，但是从概念到实践确实花了我们很长一段时间。现在的战略重点是把应用部署在 Linux 系统上。” Blanchard 说。同时，万豪酒店的系统供应商在这项系统迁移中也起到了很大的推进作用，他们并没有试图说服万豪将应用留在原来的 Unix 系统及高端服务器硬件上。“IBM 和惠普公司一直在对迁移提供帮助和支持，这使得我们的迁移工作能够顺利进行。”Blanchard 说。“我们的系统供应商对这一过渡都感到非常舒服”。但目前，基于具体情况具体分析的考虑，Unix 系统仍然是万豪复杂迁移计划的一部分。Blanchard 表示，“我们不会闭上双眼只依靠某一个特定的系统而排斥所有其它系统”。就 Qualcomm 而言，借助于 Solaris 10 系统的虚拟化技术 Solaris Containers，Qualcomm 公司获得了更多的收益。也正是 Solaris Containers 这项功能让 New York Mellon 银行取消了系统迁移计划。该银行高级工程部第一副总裁 Dennis Smith 表示，在去年 1 月份他最开始计划进行系统迁移时，他打算进行一场大规模的迁移，将银行所有的 Solaris 系统都迁移到 Linux 服务器上。不过这个想法并未完全落实。在完成了几个系统迁移之后，Smith 决定重新使用 Solaris，原因就是想要使用它的虚拟化技术，他对 Solaris Containers 进行了测试，效果不错。他说：“我们现在处于一种中间状态”。

Sun 的 Containers 技术能够创建出多个共享同一操作系统副本的虚拟机实例，Iams 认为这样可以获得规模经济效益。同时 Containers 技术的扩展性比 Vmware 要好，比 Parallels 的 Virtuozzo 更成熟。此外，Containers 还是 Sun 的核心操作系统，Sun 对它提供 全方位的支持。 “有了 Containers 系统，我们获得了更高层次的集成。使用 VMware 每个物理服务器上可能有几十个虚拟机，而采用 Containers 每台物理 机上可以安装数百个甚至数千个虚拟机。” Smith 说。

Smith 看到了 Containers 的优点而改变了他原来的大规模迁移到 Linux 系统的计划。但他依然将 Linux 纳入他的系统蓝图。“在迁移到 Linux 系统这件事上我们不会再像最初那么激进了” Smith 说。

但他还补充说：“我们认为，这两种平台将会同时出现在我们的基础设施里”。

Windows 7 能够战胜 Linux 吗？

作者：袁萌

在今年 CES 大展（元月 7 日）上微软正式推出 Windows 7 beta 测试版之后，在国际 IT 业界掀起一股邪风，叫嚷：这次与以往不同，Windows 7 一定能够战胜（crush）桌面 Linux。事实果真如此吗?

元月 9 日，Ron Barrett 发表署名文章“Why Windows 7 will crush Linux”，摆出三点理由：

1、引入命令行工具 Powershell；

2、开源软件在 Windows 平台上受欢迎；

3、性能、性能、性能。

对于第 1 点，微软引入 Powershell 命令行工具无异于承认自己的落后于 Linux；对于第 2 点，这等于说，微软已经接受了软件开发的开源模式；对于 第 3 点，实质上解决了微软的一个大难题（thorn），即利用 Aero 技术解决硬件资源的过度占用（言下之意，Windows 7 是 Vista 的“瘦身”）。实际情况是这样的呢?

元月 12 日，Linuxloop 发表专题文章“Cutting through the Windows 7 hype”（戳穿 Windows 7 的大肆宣传），指出 4 点：

1、Windows 7 仍然存在蓝屏问题；

2、整天嘀咕你打开自动更新功能；

3、易受恶意软件侵扰；

4、仍然缺少多重桌面。

根据以上所述，我们可以看出，Windows 7 并不神秘，那么玄乎。据此，凭什么说它一定能够“战胜”Linux? 更为重要的一点是，在 2006 年 6 月 28 日提出的新型 Linux 文件系统 EXT4 已 被吸收进 Linux 2.6.28 内核，特别适合固态硬盘（SSD，包括 U 盘）存储文件系统，延长 SSD 的使用寿命（EXT4 减少文件的读写次数），而在 Windows 平台上 却无此（EXT4）实现。简单说来，Linux（2.6.28 版本）更适合未来基于 SSD 的计算装置，比如“迷你本”和 MID 等。Windows 7 虽然经过“瘦身”，能够勉强塞进“迷你本”和 MID，但是，它的文件系统本质上并不适合 SSD，这是很无奈的一件事情。

近日，美国《信息世界》评论员 Randall Kennedy 言简意骇地指出：“Windows 7 looks like Vista，and it runs like Vista”，简直把微软气坏了。Vista 似乎成了“瘟疫”，用户未必喜欢，唯恐避之不及。

注 1：声称“桌面 Linux 死定了”的那个家伙，Nick Farrell(曾是演员)，发表文章“Windows 7 is enough to kill Linux on the desktop”，简直是胡扯，不值一驳。我建议大家读一下 Steven J. Vaughan-Nichols 元月 14 日发表的反驳文章“Why Linux will crush Window 7? ”，此文很值得一读。

注 2：经过验证，Ubuntu 9.04（Alpha）版本，由于使用了 EXT4 新型文件系统，系统可靠性和稳定性都大大提高，开机时间比 8.04 版本缩短 30%（用 21.4 秒钟便可“ready to go”）。

注 3：我本不想写此文，是那些 Windows 的”死硬分子”（“die hard”）先招惹起来的。

中国开源软件利用多数不规范 潜在风险巨大

近日，从“2008 中国开源软件发展峰会”获悉，中国多数企业在利用、开发开源软件时存在不规范行为，因此为企业埋下巨大的潜在风险——动辄上亿元的侵权索赔对中小企业而言不啻为“天灾”。

“在我们的调查中，还没看到没有问题的”，业界权威人士透露，保守估计，大约 70-80%的再开发开源软件都存在违反开源规则的行为。该不愿具名人士表示，主要的不规范行为是“只进不出，只用别人的开源代码，而在形成商业价值后不公开自己的源代码，这就违反了 GPL 许可证的法律约束。”

科技部知识产权事务中心副主任杨林村就开源软件可能存在的风险做了分析。“美国已有案例，GPL 合同关系受法律保护”，杨林村指出，用了他人开源软件不支出资源，在产生了商业价值后不开放自己的软件源代码是一种违约行为，应承担违约责任。同时，杨林村也指出，就目前的行业环境来说，开源软件阵营相比闭源软件尚处于相对被动的地位。

杨林村称，著作权保护的是表达方式，而不保护思想内容。如《红楼梦》再版保护的主要是文字，对软件的侵权保护主要是从表达形式保护——就是比对源代码是否 一样。其实，书写源代码只是其中一部分，如创意、商业计划书等也是体现智力成果（创意）的重要部分，而不仅仅是源代码的编写，但仅凭著作权保护显然不能起 到上诉作用。

此外，不开源的软件是否侵权判定起来非常困难，用反编译的方法取证本身就从法律上断裂了证据链条。而开源软件的侵权行为就非常容易取证，所以，起诉开源软件的侵权行为就非常容易，因此面临的诉讼的风险也最大。

更有一些企业团体为了各自利益申请了大量的垃圾专利来狙击开源软件。“95%申请的软件专利都是无效的”，杨林村呼吁健全我国软件专利审查制度和体系，防止授权垃圾专利；建立共享软件专利池及软件专利数据库，帮助开源阵营共同发展、规避侵权诉讼风险。

最后，杨林村还给出了开源软件对抗软件专利的两个策略：第一，申请专利并开放专利权；第二，不申请专利，直接公开内容，使技术丧失新颖性，从而失去可专利性。

点评 Linux 难称完美的几大命门

从 1991 年到 2008 年，Linux 已经走过了 17 个春秋，但它依然是一个正在发展中的作品，依然难称完美，还有好多方面需要完善，虽然不是致命缺陷，但是要想让 Linux 巩固现在取得的成就，并取得进一步发展，这些都需要得以解决。

软件包管理各自为政

在 linux 中，软件通过“包”形式进行管理，包可以指整个应用程序、应用程序的支持库、编程工具等等，举例来说，在多数 Linux 操作系统中，火狐浏览器和办公软件 OpenOffice.org 都是以包形式体现在其软件库中。

不同 Linux 厂商的包管理方式也有所不同。红帽使用它自己的 RPM 系统，Debian 有自己的.DEB 格式。如果你只使用某一个厂商的 Linux，这或许不是一个问题；但是当你需要跨厂商的时候，就会发现这很不方便。

这也是为什么很多商用软件厂商难于提供其产品 Linux 版的原因，没有一种统一的包格式能够克服跨厂商的问题。

面临这种情况，潜在应用软件厂商具有三种选择：一是把时间、精力和金钱用在不同 Linux 系统上，例如让自己的应用可以在红帽、SUSE 和 Ubuntu 上 安装和运行；二是只针对某一特定厂商 Linux 提供其应用；三是提供源代码包，这样用户可以在任何目标平台上自己编译代码。

第三个办法肯定不会被任何专有软件厂商所考虑。第一个办法则大大加重了应用软件厂商的工作量，基本也不可行。这样就仅仅剩下了第二个办法，既可以让用户能够迅速使用其应用程序，也降低了用户安装应用程序的工作量。

目前来看，Linux 系统上的商用软件需求还相对较少，解决这一问题的重要性还不是那么明显。但是从长远来看，当商用软件越来越多的进军 Linux 市场的时候，这无疑是 Linux 的一个很大的缺陷。一个可能的解决办法是，采用一种元包（meta-package）格式，用户下载了这种格式的文件后，使用本地软件将其处理成可以在指定系统上安装的包。目前 BitRock 有一个类似的工具，可以将一个开源应用打包成一个可在多平台上安装的程序，其中也包括对 Linux 的支持。

另一个解决此问题的主要方法是通过 Linux 标准库（Linux Standards Base，LSB）。为了兼容 LSB，Linux 厂商必须同时使用或支持红帽的 RPM。由于目前最流行的 Linux 系统是基于 Debian 的 Ubuntu，它对 RPM 的支持并不好，因此业界人士批评 LSB 过于以红帽为中心。

配置文件语法混乱

任何一个 Linux 都是多个组件和模块聚合起来的，这些软件来自成千上万个不同的程序员、项目和设计机构。这种情况导致了所有 Linux 系统都没有或很少集中配置功能，系统中的每一个模块都是通过一些杂乱无章的文件来进行设置，没有什么规定来约束和指导配置文件的语法。

如果你在工作仅仅用到少数几个配置文件，并熟悉它们的内部格式，或许不会明显的感觉到这个问题，但是这并非一个可以让人接受的解决方案。造成该问题根源是，多数应用希望保持与老的 UNIX 应用的兼容。

从内核到用户工具和应用程序，Linux 整个系统内需要一个一致的配置系统。除了便于用户（以及程序员）易于使用外，还可以简化集中管理的问题。

仅仅通过规定实现这样的事情几乎是不可能的；更好的方法是，普及推广一个可以让应用程序配置更简单的工具，从而实现统一的配置方式。GNOME 项目的 Gconf 就是这样的一个工具；尽管目前该工具的设置对象只是用户习惯设置，而并非系统范围内的配置选项，它依然为我们解决配置文件问题带来了很好的启示。

内核应用二进制接口

一直以来，在 Linux 开发领域，人们对内核应用二进制接口（Application Binary Interface，ABI）抱怨甚多。

Linux 内核设计的思路是，在内核内部可以修改很多内容，但是用户应用一定不要通过 ABI 去修改内核。这个问题不仅仅是理论性的，在实际开发中也是切实存在的：内核接口范围的存在意味着，违背其规定的某些操作完全有可能发生，有时候即使通过非常严谨的代码查阅也无法发现问题所在。

这样，当违背规定的事情发生时，它将带来两个问题：它可能让你无法确认一个问题的真正导致原因（例如它是一个内核的问题还是一个用户应用的问题？）；另外你需要花费时间和精力来修复它。

目前有一些方法来临时解决这个问题。对于某些项目来说最迅速有效的办法之一就是用户空间文件系统（Filesystem in Userspace，FUSE），它是 Linux 系统平台上可加载的内核模块，允许非

特权用户创建功能完备的文件系统，而不需要重新编译内核。FUSE 模 块仅仅提供内核模块的接入口，本身的主要实现代码位于用户空间中。但是，从长期来看，Linux 需要一个既稳定又能满足长期增长需要的 ABI，并且不会成 为造成潜在兼容性问题的老鼠窝。

原生文件版本管理（Native File Versioning ）

原生文件版本管理是另一个可以加入到 Linux 的功能，但是至今为止还没有被默认加入到 Linux 中。其概念非常简单：在一个文件当前版本被覆盖或破坏的情况下，用户可根据需要恢复到早期的任何一个版本。Windows 用户现在通过影子复制的形式可以体验这个功能，但是在标准的 Linux 文件系统中目前还没有该功能的具体体现。当然，它不能取代文件备份，但是可以把一个文件回滚到过去某个时刻的功能还是有它的用武之地的。

现在你可以手动的向 Linux 中增加这个功能。有些不同项目也已经使用略有不同的方式来实现了这个功能，诸如 Wayback、ext3cow、 copyfs 和 Tux3 等等。尽管有人称这个功能可以通过非内核插件来实现，但是如果能有一个标准的、“内核安全的”方法来实现版本控制，无疑是更好的选择。

我认为，未来的 Linux 文件系统（或许是即将到来的 BTRFS）将完全解决这个问题，但是目前还没有直接的解决方案开始解决这个问题。

音频应用程序编程接口（API）

厨师太多可能熬坏一锅好汤，用这个例子来说明 Linux 音频实现的现状再恰当不过了。多个音频 API 和子系统意味着，你可以随便选择一个来满足自己的需要，但是它也同时意味着，你将面临兼容性的问题。

内核级的音频 API，也就是 ALSA，是多数情况下应用程序的首选。但是除了它之外，还有很多其它音频 API，例如最初的 PulseAudio，主要用于混合来自多个应用程序的音频；还有 JACK，用于实现低延时的专业音频。在今年 9 月份的 Linux Plumber 大会上 Don Marti 很好的总结了该问题所带来的冲突，他表示，“如果有人来问我，‘我想编写一个音频应用程序，我应该使用哪一个 API？’我无法给出一个很好的答案。”

简而言之，音频 API 问题困扰着编程者，也困扰着用户。或者说，任何影响程序员的问题从长期来看也将影响终端用户。PulseAudio 或许是最通用的解决方案，其应用范围也最广。但是从长期来看，应用程序开发者需要的是一个内核级的音频访问方式。

图形用户界面问题

对于内核来说，需要增加什么功能自然是 Linus 和内核开发者说了算。但是对于 Linux 桌面来说，却没有什么规定可言。

在进一步阐述前，我要首先解释清楚一个概念。“桌面”不仅仅指那些让非技术用户更轻松使用 Linux/FOSS 的任何图形化用户界面，而是指一个可以让你更轻松使用和管理系统的图形化用户界面，不管你的技术能力处于哪一级别。

这不是一个将图形化用户界面完全与系统整合在一起的问题。Linux 中的内核和桌面开发是以高度并行的方式进行的，它以一个单向引导的方式进行。没有人保证内核开发者会实现对桌面开发者有用的功能，但是桌面开发却必须根据内核功能来修改自己。

这样，就需要一个指导委员会对所有运行在 Linux 上的图形化用户界面进行指导，无论创建任何图形化用户界面，不管它们是 GNOME，还是 KDE 或其它尚未发明的桌面，它们都必须具有对后端

内核功能的一致性实现，使它们能够紧密结合内核的功能。内核应该发布一个图形化用户界面可以使用的功能列表，然后图形化用户界面可以通过不同的方式来将其展现给用户。

Linux 需要做的另一件重要的事情是，需要有一个清晰的规则来指导如何使默认桌面设置更符合用户习惯，这需要设计者具有用户界面设计经验。这并不是说用户界面要“简单化”：通常需要的不是更简单的控制，而是这些控制更好的默认处理方式，这样用户无需进行麻烦的调整就可以获得最方便的设置。

X11 与应用程序的集成

大多数人用过 Linux 一段时间后，可能会遇到这样一个问题：一个 X11 应用，或 X11 自身，会出现不响应的现象，唯一的办法就是关闭并重启 X11。尽管这个操作并不复杂，但是关闭 X11 就意味着每一个其它 X11 应用都会受牵连，也会被关闭。

如果可以选择保留运行在 X 下的应用，那么当 X 需要被关掉和重启时，你就不会丢失你的整个用户会话了。如果不能实现这一点，也可以通过一个窗口管理 API 来允许轻松存储和恢复崩溃事件中的会话，这也是非常有用的一个功能。

商业化备份恢复解决方案支持

如果 Linux 希望继续吸引普通的 PC 用户从 Windows 转向自己，它需要具有 Windows 中的许多类似功能。其中包括支持 Linux 客户端的付费网络服务，诸如远程备份等。

Windows 和 Mac 用户具有众多选择：这两个系统中的本地文件和系统级别的备份和恢复都可以通过众多商业化备份解决方案所实现。诸如 Mozy 和 Carbonite 之 类的服务可以进行安静、加密的差异化备份到远程主机，因此用户的数据可以被连续保护并离线保存。但是这些服务都不支持 Linux 客户端。

Linux 用户如果希望执行离线备份，通常要借助于本地 Linux 应用程序，诸如 Ubuntu 中的简单备份工具，来备份到一个并非专门用于数据保护的远程服务器。实际上并不缺少可以与远端服务器进行会话的 Linux 备份客户端，例如在 Ubuntu 中就有简单备份套件，但是对普通用户来说，却缺少一个与商业化备份提供商整合的应用。

一种可能发生的事情是，随着一切变得与平台无关，所有数据的备份都将可以通过特定 Web 浏览器来实现。另一种更可能发生的事情是，专门支持 Linux 客户端的完整状态备份服务将会出现。

结论

本文中所提到的所有 Linux 缺陷都不是致命的，否则 Linux 就不会取得今天的成绩。但是毫无疑问，这些问题都需要被改进，这可能需要改变某些传统的 Linux 功能设计思路，只有保持不断革新、去伪存真，才能让 Linux 真正成为一件完美的产品。

技术沙龙

IT168 技术精英年会 CU 论坛回眸

2009 年 1 月 9 日下午，北京新世纪日航饭店，IT168 技术精英年会技术分论坛如期举行。通过 ChinaUnix 论坛报名的有《开源新时代》和《网络优化》两个分论坛。真诚地感谢大家的积极捧场，本次技术论坛规模空前，参加这两个论坛的人数超过 400 人。

《开源新时代》论坛主要邀请了国内知名的开源专家学者、企业总裁和创业代表，如倪光南院士、中科红旗副总郑忠源、红旗 2000 总经理胡才勇和 Discuz 创始人戴志康等。上述各位代表的是国内开源业界的最高水平，各位大师侃侃而谈，让大家把握住了最新的开源行业趋势。而《网络优化》论坛则是云集了国内各大网站、以及传统企业的运维工程师，例如有的来自新浪、搜狐、百度等业内巨头。《云计算》和《虚拟化》两个分论坛则是本次 ChinaUnix 技术论坛顺应目前 IT 趋势新推出的沙龙主题，同样得到了厂商和网友的热烈欢迎。

部分参会嘉宾



中国科学院院士：倪光南



Intel 中国开源战略经理：陈绪



中科红旗副总裁：郑忠源



IBM Tivoli 开发经理：谭瑞忠



Google 开发工程师：王超



思杰网络技术顾问：李兵

部分会场照片

网络分论坛



踊跃交流



与会者合影留念



会场座无虚席

开源新世纪论坛



签到处


签到场景


圆桌会议

虚拟化论坛


分论坛主持人洪钊峰



演讲嘉宾张哲在演讲



IT168 网站副总编辑：江晖

云计算论坛


樊强在抽奖



用户会场外交流


听众认真听讲

部分演讲资料

网络优化分论坛

曹刚：门户网站系统运维架构规划设计实战

吴炳锡：LVS、Nginx 负载分衡构建实战

梁世鹏：Radware 梳理业务的脉络神经

叶金荣：Linux 下的 MySQL 调优

田逸：多数据中心站点 CDN 网络构建实例

云计算分论坛

云计算时代下的企业 IT 运维变迁

在云之上构建企业数据中心

利用云计算技术提升企业服务水平

开源新时代分论坛

倪光南：开源软件对于中国的意义

邵宗文：如何利用 MySql 架构大型应用

程勇：基于 Terracotta 的可扩展集群架构

张翔：Mysql 数据库性能优化应用

虚拟化分论坛

服务器虚拟化调查报告

虚拟化实战经验案例分析

红帽企业虚拟化方案剖析

红帽 Linux 5 虚拟化实战

利用应用虚拟化技术搭建安全的研发中心

技术新知

独辟蹊径网络安装系列之 Debian/Ubuntu

ChinaUnix 网友：kns1024wh

此文章是 Linux 部署方式系列文章中的第二部分，探讨开源社区的热点 Debian/Ubuntu 的所特有的软件源问题，以及实现网络安装方式的独特之处。所谓 Debian/Ubuntu 软件源就是一个应用程序安装库，很多很多的应用软件都在这个库里面。可以是网络服务器，是光盘，甚至是硬盘上的一个目录。作为 Debian 系的 Ubuntu,继承了 Debian 的 deb 和 apt 系统，只要设定好软件源，就能很方便的安装软件了以及实现从网络安装 Debian/Ubuntu 系统本身。从实现的原理上可以将 DVD/CD 的介质使用 dpkg、apt-move 等命令工具结合本地的 http、ftp 服务可以实现一个本地的源，当此种方式生成的源多数能够完成软件安装工作，对于 Debian/Ubuntu 基于互联网的实时更新的特性就不能够很好的发挥出来，重要的一点是对从 PXE 引导的网络安装支持很不理想，对于维护数量众多的 Debian/Ubuntu 服务器来说不是很好的选择。

本文将讲述基于同步镜像 Debian/Ubuntu 某个版本一个官方的镜像的方式，实现一个基于局域网本地的 Debian/Ubuntu 源的网络安装部署 Debian/Ubuntu 服务器的方式，当然并不是全部镜像，故此只能实现镜像版本的网络安装，此方式已经足够满足网络安装的要求。

Debian 官方站点 http://www.debian.org

Ubuntu 官方站点 http://www.ubuntu.com

开始工作工作之前先了解 Debian/Ubuntu 发行版本的渊源，Debian 的开发代号来源于电影《玩具总动员》，而脱胎于 Debian 的 Ubuntu，其开发代号同样很有意思。除前两个版本之外，开发代号命名按字母顺序排列，在动物名之前按照双重字母再选个形容词:Warty Warthog, Hoary Hedgehog, Breezy Badger, Dapper Drake, Edgy Eft, Feisty Fawn, Gutsy Gibbon, Hardy Heron.

表：Debian/Ubuntu 发行版本信息，按照发行的先后顺序列出

Debian 发行版	Ubuntu 发行版
Debian GNU/Linux 2.0 (hamm)	Ubuntu 4.10 - Warty Warthog(长疣的疣猪)
Debian GNU/Linux 2.1 (slink)	Ubuntu 5.04 - Hoary Hedgehog(灰白的刺猬)
Debian GNU/Linux 2.2 (potato)	Ubuntu 5.10 - Breezy Badger(活泼的獾)
Debian GNU/Linux 3.0 (woody)	Ubuntu 6.06 - Dapper Drake(整洁的公鸭)
Debian GNU/Linux 3.1 (sarge)	Ubuntu 6.10 - Edgy Eft(急躁的水蜥)
Debian GNU/Linux 4.0 (etch)	Ubuntu 7.04 - Feisty Fawn(坏脾气的小鹿)

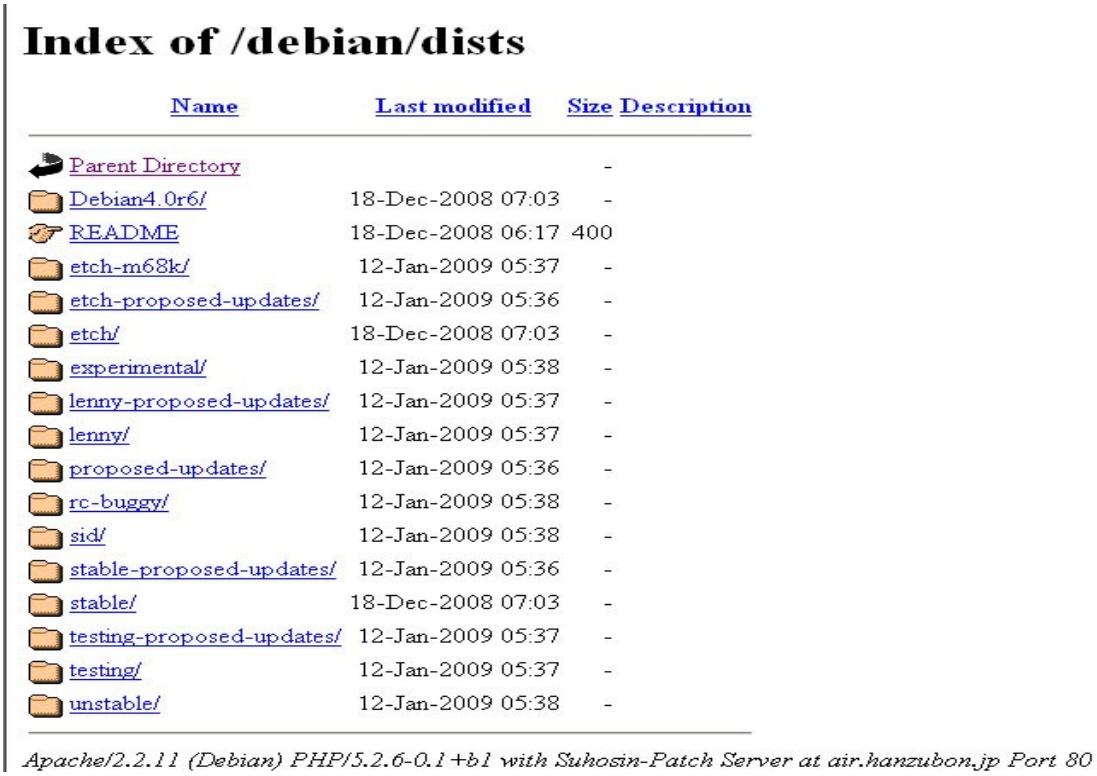
下一代 Debian 正式发行版的代号为 lenny	Ubuntu 7.10 - Gutsy Gibbon(勇敢的长臂猿)
	Ubuntu 8.04 - Hardy Heron （耐寒的苍鹭）
	Ubuntu 8.10 -Intrepid Ibex（无畏的北部高地山羊）
	Ubuntu 9.04-Jaunty Jackalope（活泼的怀俄明野兔）

当访问 Debian/Ubuntu 的官方源镜像站点是会在 dists/目录下看到发行版本代号的目录名称。镜像 Deiban/Ubuntu 的源中的某个版本也是依据此目录中的代号为依据的。

当前 Debian 全球镜像站点清单 http://www.debian.org/mirror/list;

当然 Ubuntu 全球镜像站点清单 https://launchpad.net/ubuntu/+archivemirrors。

获知这些信息对于镜像某个发行版本或者是修改系统的 sources.list 都是可以自行决定的，当然建立了一个自己的源将会更加的便利。补充一下 Debian/Ubuntu 的镜像都有 Archive Mirror 和 Ubuntu releases 之分，前者就是本文所说的软件源，或者就是通常所说的 ISO 格式的发行光盘介质。



Index of /debian/dists

Name	Last modified	Size	Description
Parent Directory		-	
Debian4.0r6/	18-Dec-2008 07:03	-	
README	18-Dec-2008 06:17	400	
etch-m68k/	12-Jan-2009 05:37	-	
etch-proposed-updates/	12-Jan-2009 05:36	-	
etch/	18-Dec-2008 07:03	-	
experimental/	12-Jan-2009 05:38	-	
lenny-proposed-updates/	12-Jan-2009 05:37	-	
lenny/	12-Jan-2009 05:37	-	
proposed-updates/	12-Jan-2009 05:36	-	
rc-buggy/	12-Jan-2009 05:38	-	
sid/	12-Jan-2009 05:38	-	
stable-proposed-updates/	12-Jan-2009 05:36	-	
stable/	18-Dec-2008 07:03	-	
testing-proposed-updates/	12-Jan-2009 05:37	-	
testing/	12-Jan-2009 05:37	-	
unstable/	12-Jan-2009 05:38	-	

Apache/2.2.11 (Debian) PHP/5.2.6-0.1+b1 with Suhosin-Patch Server at air.hanzubon.jp Port 80

图：浏览 Debian 软件源 dists 目录 http://air.hanzubon.jp/debian/dists/



Name	Last modified	Size	Description
Parent Directory		-	
dapper-backports/	12-Nov-2008 08:00	-	
dapper-proposed/	12-Nov-2008 08:00	-	
dapper-security/	12-Nov-2008 08:00	-	
dapper-updates/	12-Nov-2008 08:00	-	
dapper/	16-Dec-2008 17:19	-	
feisty-backports/	12-Nov-2008 08:00	-	
feisty-proposed/	12-Nov-2008 08:00	-	
feisty-security/	12-Nov-2008 08:00	-	
feisty-updates/	12-Nov-2008 08:00	-	
gutsy-backports/	12-Nov-2008 06:13	-	
gutsy-proposed/	10-Jan-2009 07:38	-	
gutsy-security/	12-Nov-2008 08:00	-	
gutsy-updates/	12-Nov-2008 08:00	-	
gutsy/	12-Nov-2008 08:00	-	
hardy-backports/	12-Nov-2008 08:00	-	
hardy-proposed/	12-Nov-2008 08:00	-	
hardy-security/	12-Nov-2008 08:00	-	
hardy-updates/	12-Nov-2008 08:00	-	
hardy/	12-Nov-2008 08:00	-	
intrepid-backports/	10-Jan-2009 07:38	-	
intrepid-proposed/	22-Nov-2008 06:17	-	
intrepid-security/	26-Nov-2008 06:15	-	
intrepid-updates/	13-Nov-2008 06:32	-	
intrepid/	12-Nov-2008 08:00	-	
jaunty-backports/	12-Nov-2008 08:00	-	
jaunty-proposed/	12-Nov-2008 08:00	-	
jaunty-security/	12-Nov-2008 08:00	-	
jaunty-updates/	12-Nov-2008 08:00	-	
jaunty/	12-Nov-2008 06:13	-	

Apache/2.2.9 (Unix) Server at de.archive.ubuntu.com Port 80

图：浏览 Ubuntu 源 dists 目录 http://de.archive.ubuntu.com/ubuntu/dists/

镜像一个 Debian/Ubuntu 源的方式有很多，经过尝试排除了 apt-mirror 、debmirror 方式，选择 rsync 方式这个是在 Debian/Ubuntus 上都是适用的镜像发行版的方法。笔者在测试过程中适用的带宽仅为 4MB，镜像 Debian 的 etch 发行版本用时大约是 10 天左右，镜像 Ubuntu 的 intrepid 版本用时大概 2 天时间，磁盘空间都是占用非常大的，如果没有 500GB 的磁盘容量和大于 10MB 的线路连接建议还是不要轻易尝试。目前无论是 Debian 还是 Ubuntu 官方站点都没有名且说明同步某一个或几个特定版本的方式，通过 rsync 进行同步可以在使用的时候首先要访问进行站点的 dists 目录，根据该站点使用--exclude 排除具体的版本，格式：--exclude=* “发行版本的代号，如 intrepid” * --delete-excluded 把这个加入你的参数中即可（注意有两个星号，不能少呀）。以下是测试过的 Debian/Ubuntu 的同步脚步，进攻参考。

表：Ubuntu rsync 同步参考脚步

```
#!/bin/bash
HOST=de.archive.ubuntu.com
MIRROR_ROOT='ubuntu'
LOCAL="/usr/src/o"
OPTIONS="-vzrtopglK --progress --delete --delete-excluded"
EXCLUDE="--exclude daily-installer-powerpc/ \
--exclude installer-powerpc/ \
--exclude binary-powerpc/ \
--exclude upgrade-powerpc/ \
--exclude disks-powerpc/ \
--exclude *_powerpc.udeb \
--exclude *_powerpc.deb"

rsync $OPTIONS $EXCLUDE $HOST::$MIRROR_ROOT $LOCAL
```

表：Debian rsync 同步参考脚步

```
#!/bin/bash
HOST=ftp.jp.debian.org
SRC='debian'
DST="/usr/src/o"
OPTIONS="-aPS --delete-during --delete-excluded"
EXCLUDE="--exclude *alpha/ --exclude *_alpha.deb --exclude Contents-alpha* \
--exclude *arm/ --exclude *_arm.deb --exclude Contents-arm* \
--exclude *hppa/ --exclude *_hppa.deb --exclude Contents-hppa* \
--exclude *ia64/ --exclude *_ia64.deb --exclude Contents-ia64* \
--exclude *m68k/ --exclude *_m68k.deb --exclude Contents-m68k* \
--exclude *mips/ --exclude *_mips.deb --exclude Contents-mips* \
--exclude *mipsel/ --exclude *_mipsel.deb --exclude Contents-mipsel* \
--exclude *powerpc/ --exclude *_powerpc.deb --exclude Contents-powerpc* \
--exclude *s390/ --exclude *_s390.deb --exclude Contents-s390* \
--exclude *sparc/ --exclude *_sparc.deb --exclude Contents-sparc* \
--exclude *sarge* --exclude *Debian3* --exclude *oldstable* \
--exclude *.iso \
--exclude *~ \
```

```
--exclude *.orig.tar.gz --exclude *.diff.gz --exclude *.dsc"
rsync $OPTIONS $EXCLUDE $HOST::$SRC $DST
```

将所需要的发行版本同步完成后，只需要在现有的 apache 的 documentroot 目录下面建立一个软连接如：ln -s /usr/src/o /var/www/html/ubuntu 就可以通过浏览器访问到本地源。当然也可以使用 ftp 的方式，ftp 在穿越 NAT 以及防火墙的时候的策略问题，所以不推荐 ftp 模式。

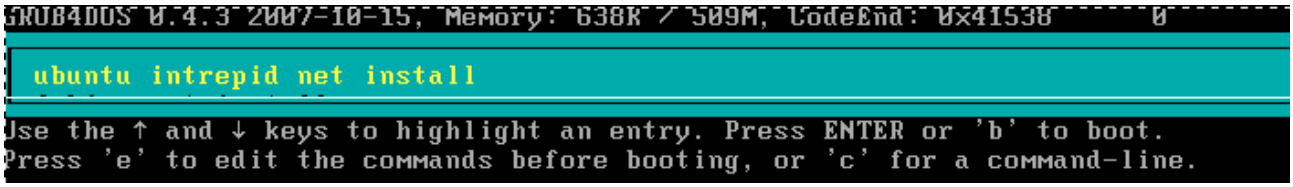
以下将讲解，在 windows 环境测试 Debian/Ubuntu 网络安装过程，进行网络安装需要选择的适合的内核引导文件 initrd.gz 和 linux ，均需要下载本地镜像的 netboot.tar.gz。

下载 Ubuntu 的网络引导文件 http://192.168.1.1:11580/ubuntu/dists/intrepid/main/installer-i386/current/images/netboot/boot.img.gz
下载 Debian 的网络引导文件：http://192.168.1.1:11580 /debian/dists/etch/main/installer-i386/current/images/netboot/netboot.tar.gz

将下载的 netboot.tar.gz 在 windows 系统的 C 盘的根目录下面，（如果是启用 PXE 安装只需要将 netboot.tar.gz 文件解压到/tftpboot 目录中，此内容将在后续文章讲解）确认系统已经安装 grub for dos 然后编辑 menu.list 内容如下：

```
title ubuntu intrepid net install
root (hd0,0)
kernel /linux root=/dev/ram ramdisk_size=256000 devfs=mount,dall
initrd /initrd.gz
boot
```

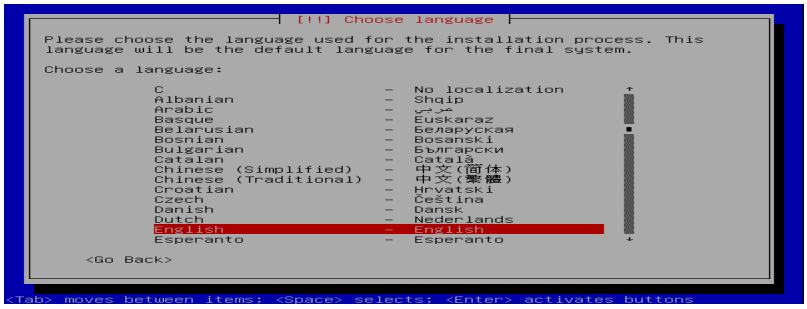
这样重新开机引导就会进入 Debian/Ubuntu 引导过程



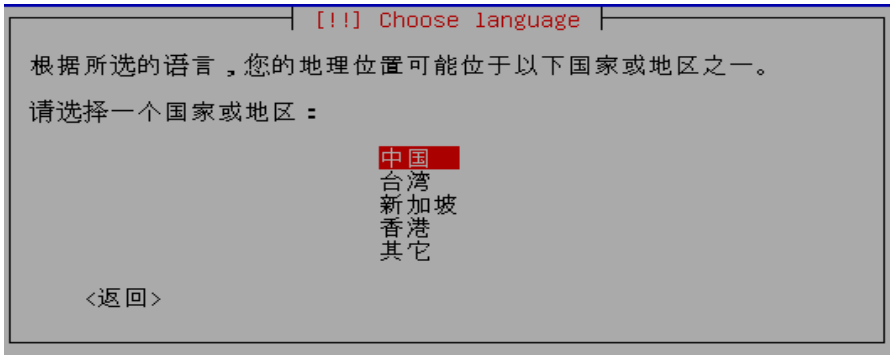
图：Ubuntu intrepid net install Grub 引导选项

因 debian/Ubuntu 版本的渊源其网络引导过程极为类似，但 Ubuntu 提供了更多衍生版本的安装选择如 Xubuntu desktop、Kubuntu desktop、Edubuntu desktop、Myehuntu 、Ubuntu mobile、Ubuntu MID edition、Basic Ubuntu Server 等众多的选择。故此本文的安装过程以演示 Ubuntu intrepid 版本网络安装的过程为主。

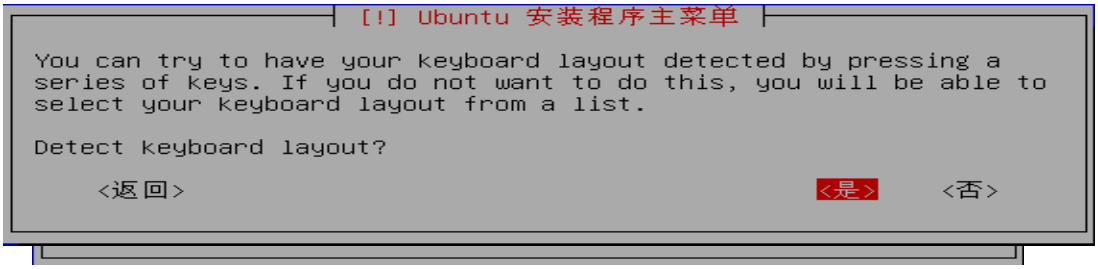
在系统初始化引导结束后将进入语言选择，并开始设置安装的选项与参数。



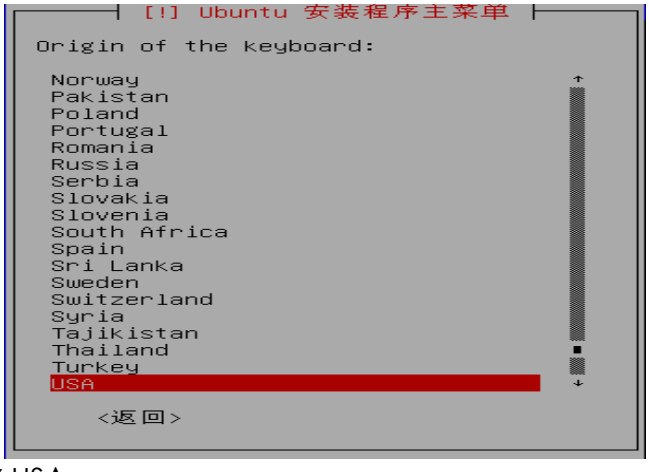
图：语言选择，本文选择中文（简体）



图：语言选择，本文选择中国



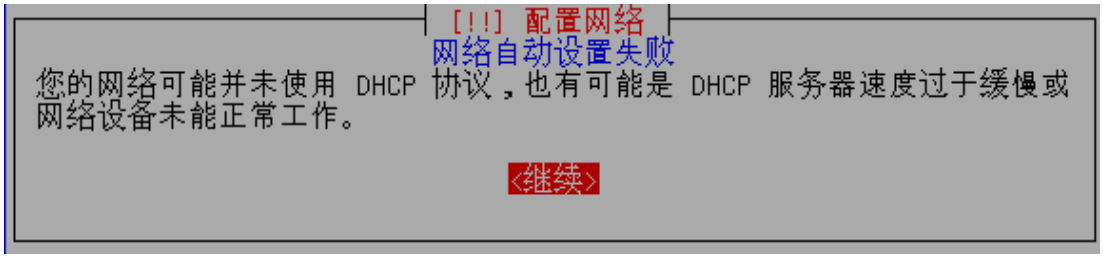
图：语言支持提示，选择否继续图：键盘选择，时候使用自动检测键盘类型，选择否继续



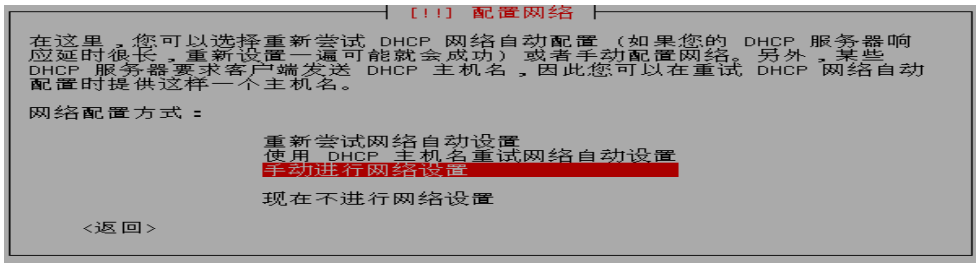
图：键盘选择提示，选择 USA



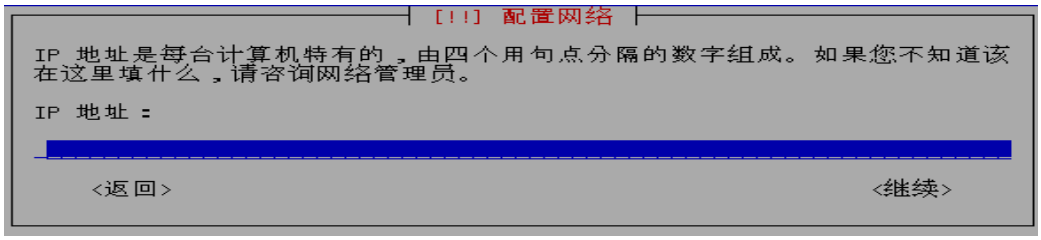
图：键盘选择提示，选择 USA



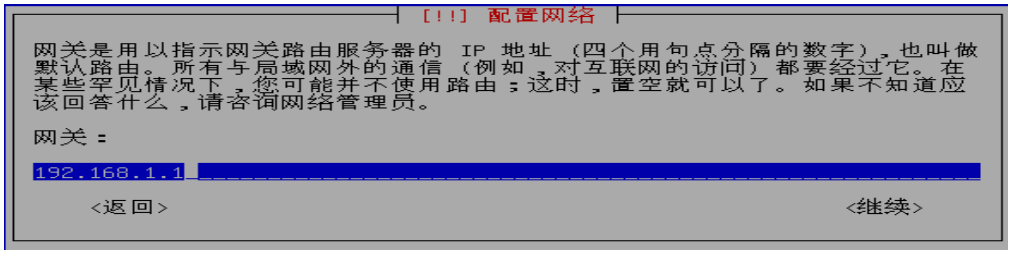
图：网络检测，网卡识别，无 DHCP，选择继续执行下部手动配置



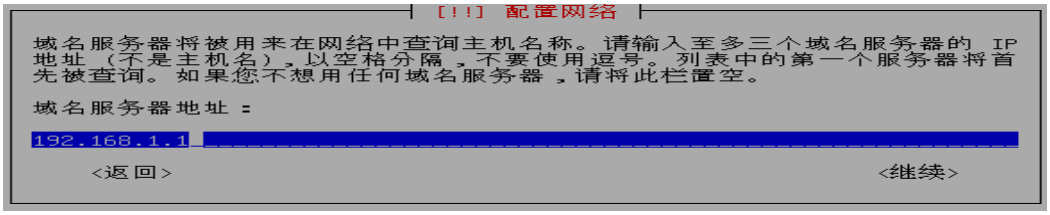
图：网络检测，选择手动进行网络设置（这个是必须的）



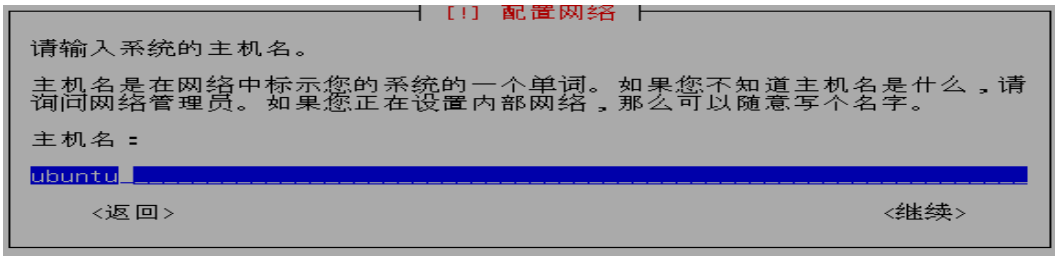
图：设置 IP 地址及掩码信息



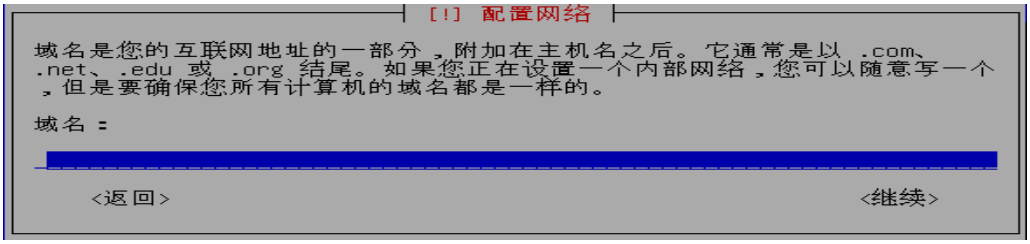
图：设置网关信息



图：设置 DNS 服务器信息

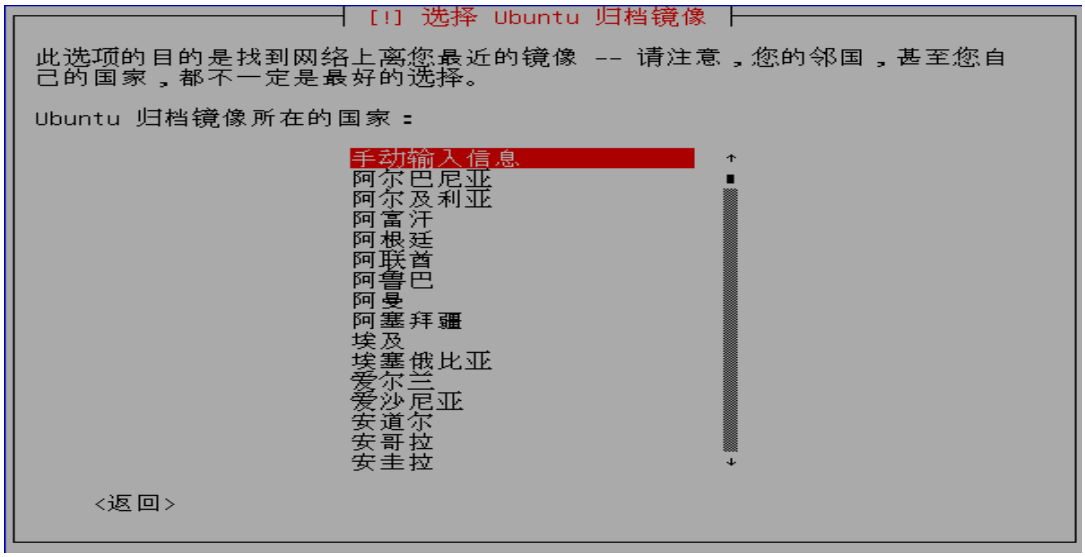


图：设置主机名称，默认为 ubuntu 或者 debian

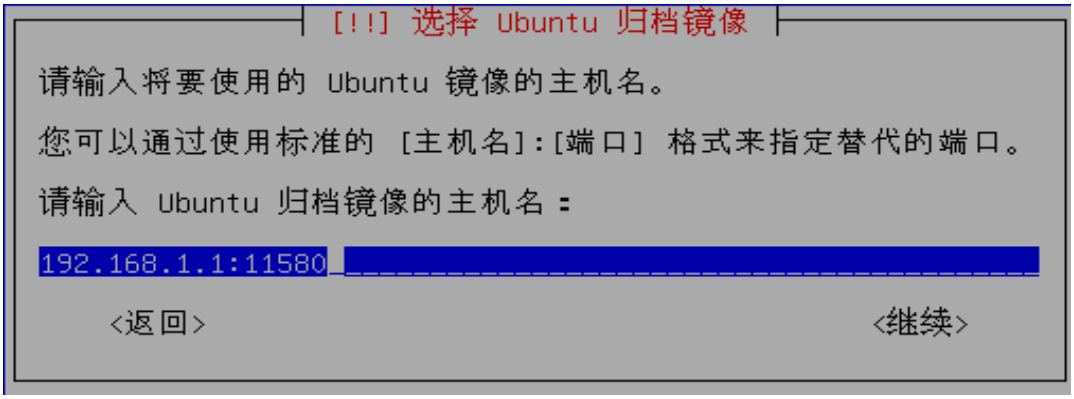


图：设置域名信息，如果局域网没有本地 DNS 默认为空

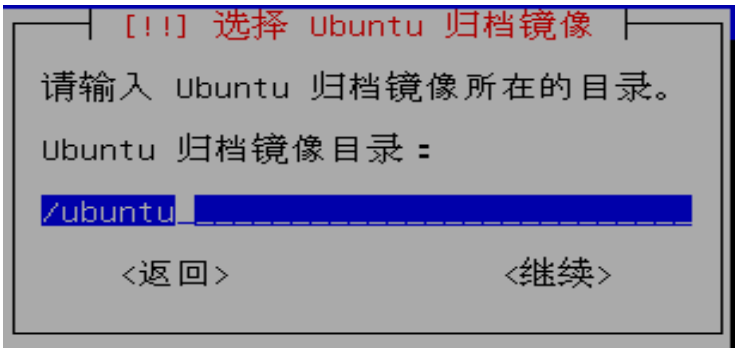
以上步骤完成了 Debian/Ubuntu 安装的基本设置，下一步骤就是要设置使用本地的已经建立好的软件源，进行必要安装选项设置以及必要的文件复制和系统默认配置。以上步骤使用的是网络安装的引导内核，在设置完毕软件源将会下载基本的系统到本地运行。



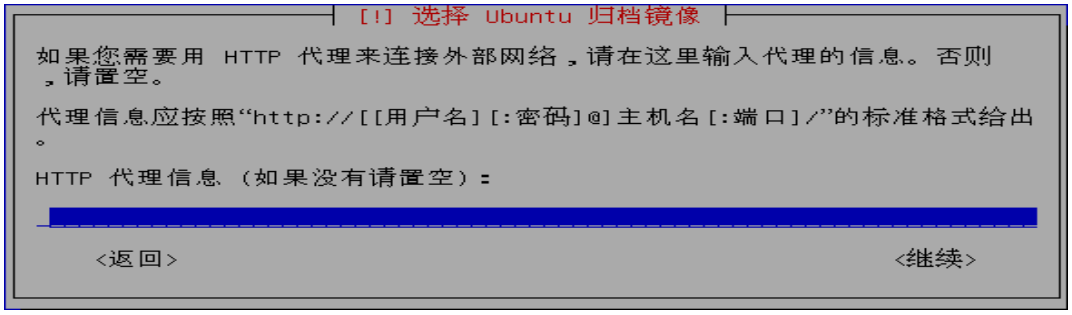
图：选择 Debian/Ubunut 安装源，选择手动输入信息



图：输入进行服务器的 IP 地址和端口，默认 80 端口无需输入

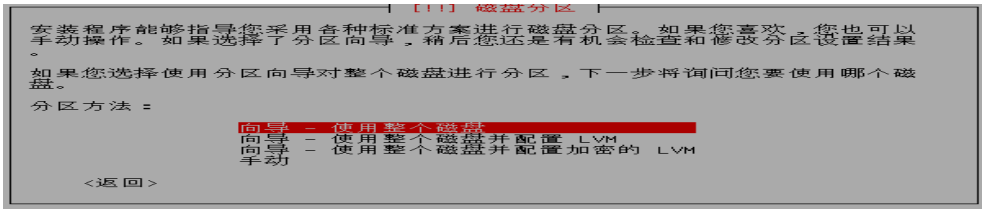


图：输入镜像目录名称，默认为/debian/或/ubunut/ 请将最后的/符合删除

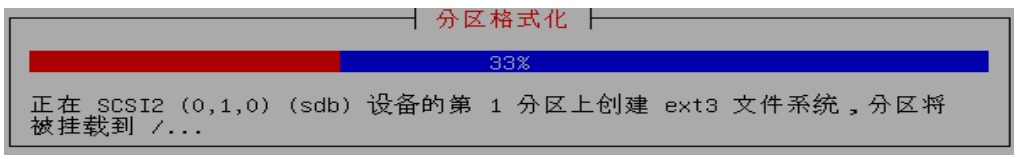


图：设置 web 代理服务器信息

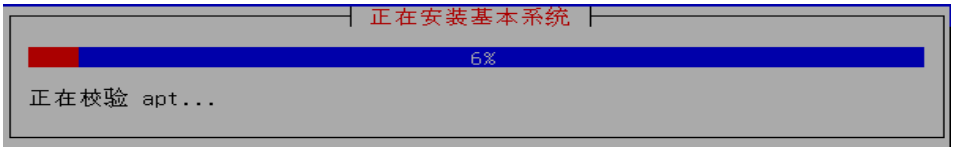
此步骤如果看到的信息是不全的或者是以- 、空格等符号显示，说明镜像制作的有问题，需要重新同步。设置好系统的分区后就开始网络安装过程，观察镜像服务器的网卡流量将会增长较大。



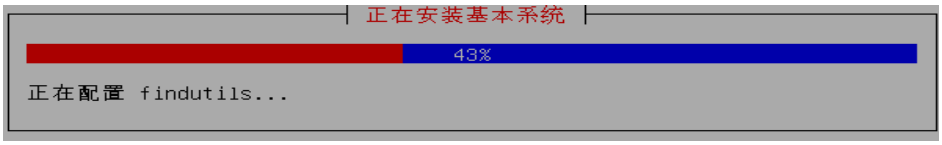
图：设置系统分区



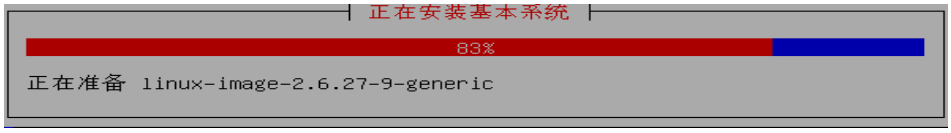
图：分区格式化



图：安装基本系统



图：安装基本系统

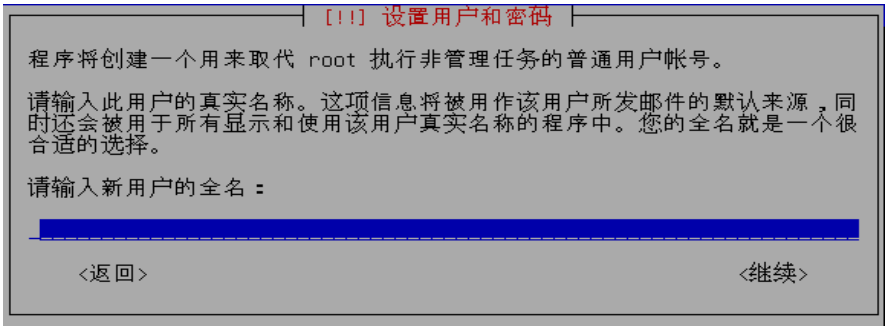


图：安装基本系统

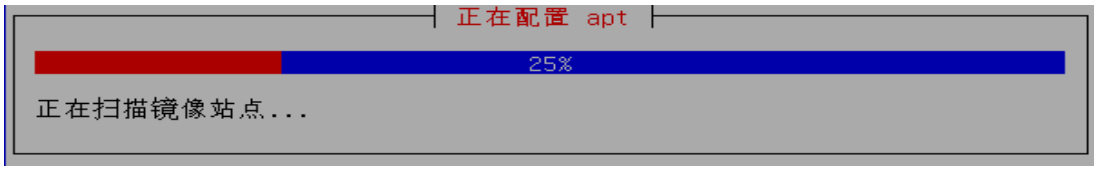
```
Jan 10 07:52:24 kernel:  sdb1
Jan 10 07:52:24 kernel: [  310.047157] sd 1:0:1:0: [sdb] Attached SCSI disk
Jan 10 07:52:24 kernel: [  310.048264] sr0: scsi3-mmc drive: 1x/1x xa/form2 cdda
 tray
Jan 10 07:52:24 kernel: [  310.048368] Uniform CD-ROM driver Revision: 3.20
Jan 10 07:52:24 kernel: [  310.048600] sr 2:0:0:0: Attached scsi CD-ROM sr0
Jan 10 07:52:25 hw-detect: Loading PCMCIA bridge driver module: i82365
Jan 10 07:52:25 kernel: [  310.479159] Intel ISA PCIC probe: not found.
Jan 10 07:52:26 apt-install: Queueing package eject for later installation
Jan 10 07:52:26 net/hw-detect.hotplug: Detected hotpluggable network interface e
th0
Jan 10 07:52:27 net/hw-detect.hotplug: Detected hotpluggable network interface l
o
Jan 10 07:52:27 anna-install: Installing dmraid-udeb
Jan 10 07:52:27 anna[15662]: DEBUG: retrieving dmraid-udeb 1.0.0.rc14-2ubuntu12
Jan 10 07:52:28 anna[15662]: DEBUG: retrieving libdmraid1.0.0.rc14-udeb 1.0.0.rc
14-2ubuntu12
Jan 10 07:52:28 kernel: [  313.247120] device-mapper: uevent: version 1.0.3
Jan 10 07:52:28 disk-detect: insmod /lib/modules/2.6.27-7-generic/kernel/drivers
/md/dm-mod.ko
Jan 10 07:52:28 kernel: [  313.249230] device-mapper: ioctl: 4.14.0-ioctl (2008-
04-23) initialised: dm-devel@redhat.com
Jan 10 07:52:28 disk-detect: No Serial ATA RAID disks detected
Jan 10 07:52:28 main-menu[2610]: DEBUG: resolver (libnewt0.52): package doesn't
exist (ignored)
Jan 10 07:52:28 main-menu[2610]: DEBUG: resolver (efi-modules): package doesn't
exist (ignored)
Jan 10 07:52:28 main-menu[2610]: INFO: Falling back to the package description f
or console-setup-udeb
```

图：查看到系统安装日志信息 /var/log/syslog

在安装过程，可以通过 ALT+F2 切换到其它终端窗口查看系统的安装日志信息，可以通过此日志判断问题所在。日志查看命令 tail -f /var/log/syslog



图：设置用户和密码，这就是有名的 sudo 的开始



图：扫描镜像站点，准备下一步的软件配置



图：自动更新选项（根据网络状况选择）

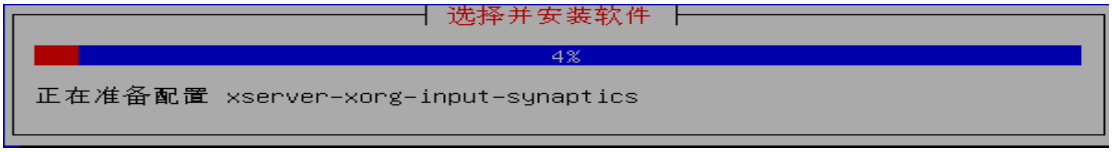


图：软件选择

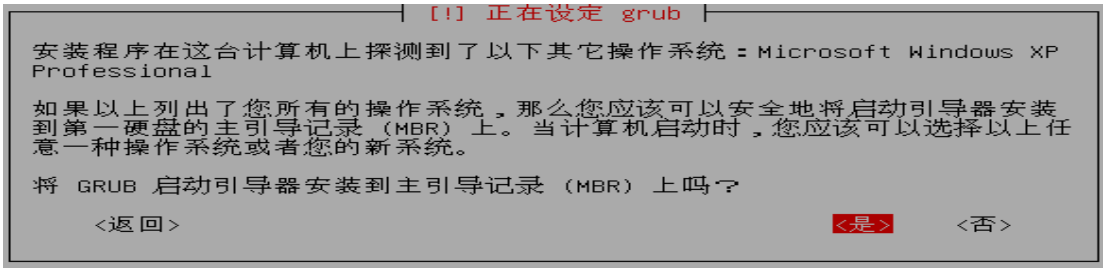


图：软件选择

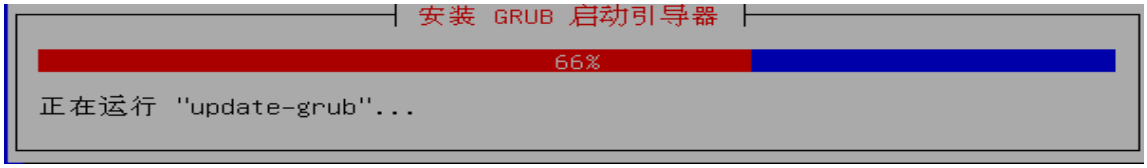
上述两图是 Debian 与 Ubuntu 版本明显区别的地方，总体感觉 Ubuntu 的选择会比 Debian 有很多的便利。如果不使用软件源安装是看不到如此多的软件选择的。



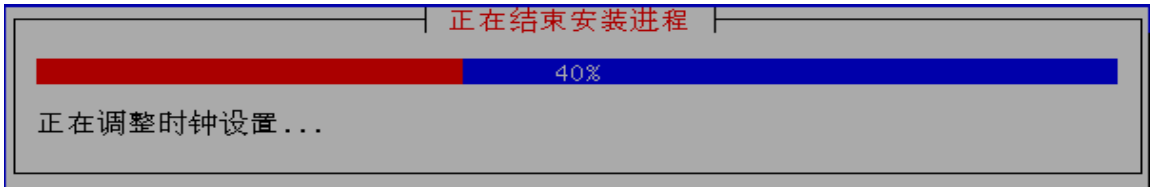
图：软件安装过程



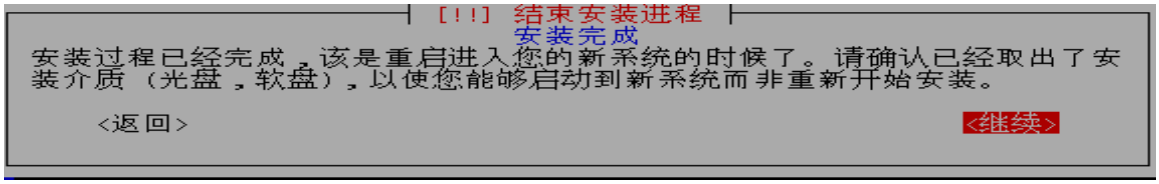
图：设置 grub 提示，选择是



图：安装 Grub 过程



图：设置系统时间，完成最后安装配置



图：Debian/Ubuntu 安装完成



图：启动过程中的 Ubuntu Logo



图：工作中的 Ubuntu 桌面

到此 Debian/Ubuntu 通过局域网本地的源进行安装的过程已经全部结束。从整个过程中看，主要问题点在于镜像的同步实现，如果没有足够的带宽这个工作就没有保障性，就会出现失败问题。如果企业使用 Debian/Ubuntu 作为服务器版本合理的方式就是要在局域网内部配置一个定期同步的本地源，这个方式希望对 Linux 用户能够有所提示和启发，欢迎就相关问题进行交流。

作者简介：CU 网友 kns1024wh，目前从事 Linux 群集方面的具体工作，之前做过多年的 IT 技术支持、MCT 讲师、及 REDFLAG 的技术合作，技术专长群集、unix 主机、AD 部署等，您可以通过电子邮件 lvsheat@qq.com 或者 Chinaunix 社区与他取得联系。

初探 Cherokee：号称最快的 web 服务器

ChinxUnix 网友：eScaPedd

习惯了 Apache、lighttpd、nginx，不知道你用过 cherokee 这个 web 服务器没。“cherokee 比 nginx 还快”，相 信这个理由足够让你来尝试一下这个目前号称最快的 web 服务器。经过一番尝试，发现 cherokee 还有一个很爽的功能：图形化的 web 管理界面（类似 zeus 的管理界面），怎么样，心动了吧。

cherokee 的官方网址是：http://www.cherokee-project.com，你可以从这里下载到最新的源代码，找到它详细的说明文档。

下面这段文字简单翻译自官方的说明文档（英文很烂，见笑了）

cherokee 是一个高效的、轻量级的、高稳定性的、容易配置的 web 服务器 …
cherokee 支持很多技术：FastCGI , SCGI , PHP, CGI , X-Sendfile , TLS , SSL …
cherokee 支持虚拟主机、权限认证、负载均衡 …
cherokee 的日志格式与 apache 是兼容的 …
cherokee 可以在不中断服务的前提下进行升级更新（nginx 也有这个功能）…

到今天为止，cherokee 的最新版本是 0.11.6 了。

说明：以下所有操作都是在我用 VMWare 虚拟的一个 CentOS 下以 root 用户进行的。

下载了最新的源代码过后，照常，编译安装：

```
./configure --prefix=/usr/local/cherokee

make

make install
```

编译过程非常简单，也非常顺利。cherokee 的 configure 脚本可以传入一个 PHPCGI 的环境变量，用来设置 php-cgi 的路径，比如这样:

```
PHPCGI=/usr/local/php5/bin/php-cgi ./configure --prefix=/usr/local/Cherokee
```

当然，如果你的 php-cgi 在系统 PATH 环境变量下，也就不用设置了，或者编译好以后再去修改配置文件也来得及。

cherokee 另外一个让人兴奋的特性是，它自己带了一个图形化的 web 管理端，就像 zeus 一样。通过这个图形化的管理端，你可以对 cherokee 进行几乎所有的日常配置、管理操作，非常非常方便。

你可以通过下面的步骤启动 cherokee 的图形化管理端：

```
cd /usr/local/cherokee/sbin

./cherokee-admin -b 172.16.236.248
```

其中 cherokee-admin 的-b 参数是用来控制管理端监听的端口的，默认只监听 127.0.0.1，你要想通过局域网中的其他计算机访问这个管理端，那么就把我写的 172.16.236.248 换成 cherokee 所在服务器的局域网 IP 地址。

从命令行启动 cherokee 管理端以后，你会在命令行看到类似这样的信息：

Login:

User: admin

One-time Password: mXvZEUYfpupeR9o3

Cherokee Web Server 0.11.4 (Dec 18 2008): Listening on port 9090, TLS disabled,

IPv6 disabled, using epoll, 1024 fds system limit, max. 505 connections,

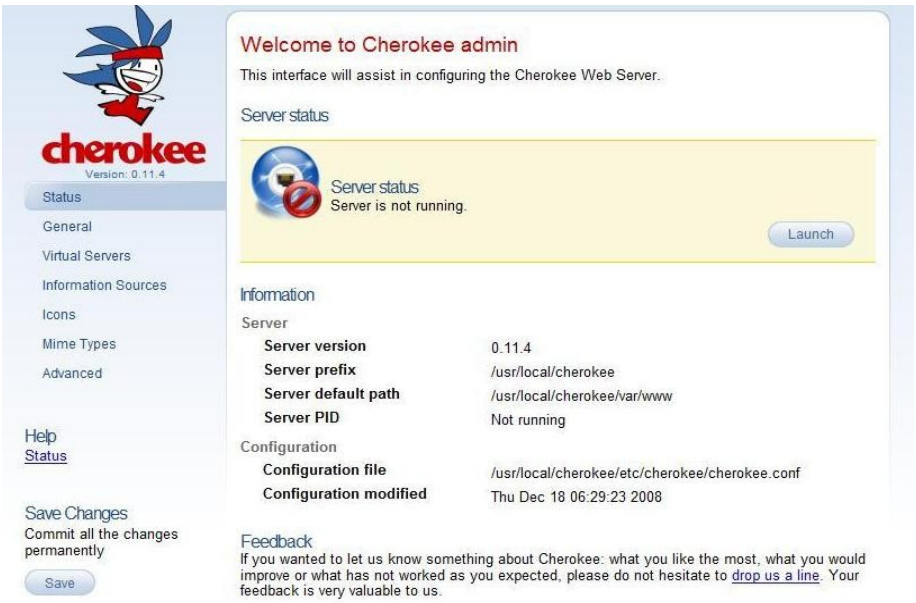
single thread

上面的信息告诉你：

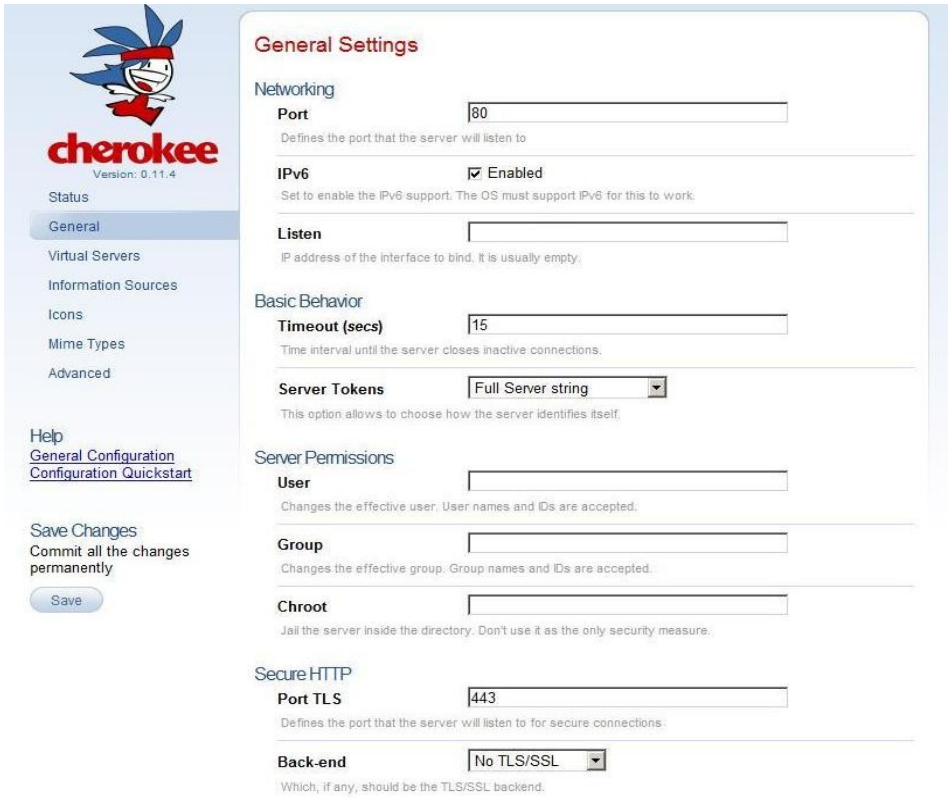
1、cherokee 的管理端监听在服务器的 9090 端口（和 zeus 默认的一样）
2、每次启动 cherokee-admin，系统都会生成一个一次性的登录密码，你只有使用这个密码通过管理端的 http auth basic 以后才能进行相关操作
3、其他的服务器信息，比如版本号、最多允许的连接数等等

在局域网的另外一台计算机中打开一个浏览器，输入 http://172.16.236.248:9090 ，你将会看到 cherokee-admin 的登录认证窗口，输入刚才命令行提示的一次性密码，你就可以看到这个让人兴奋的图形化管理界面了。

默认的 Status （服务器状态，从这里可以看到 cherokee 的基本信息，并启动、停止 cherokee）



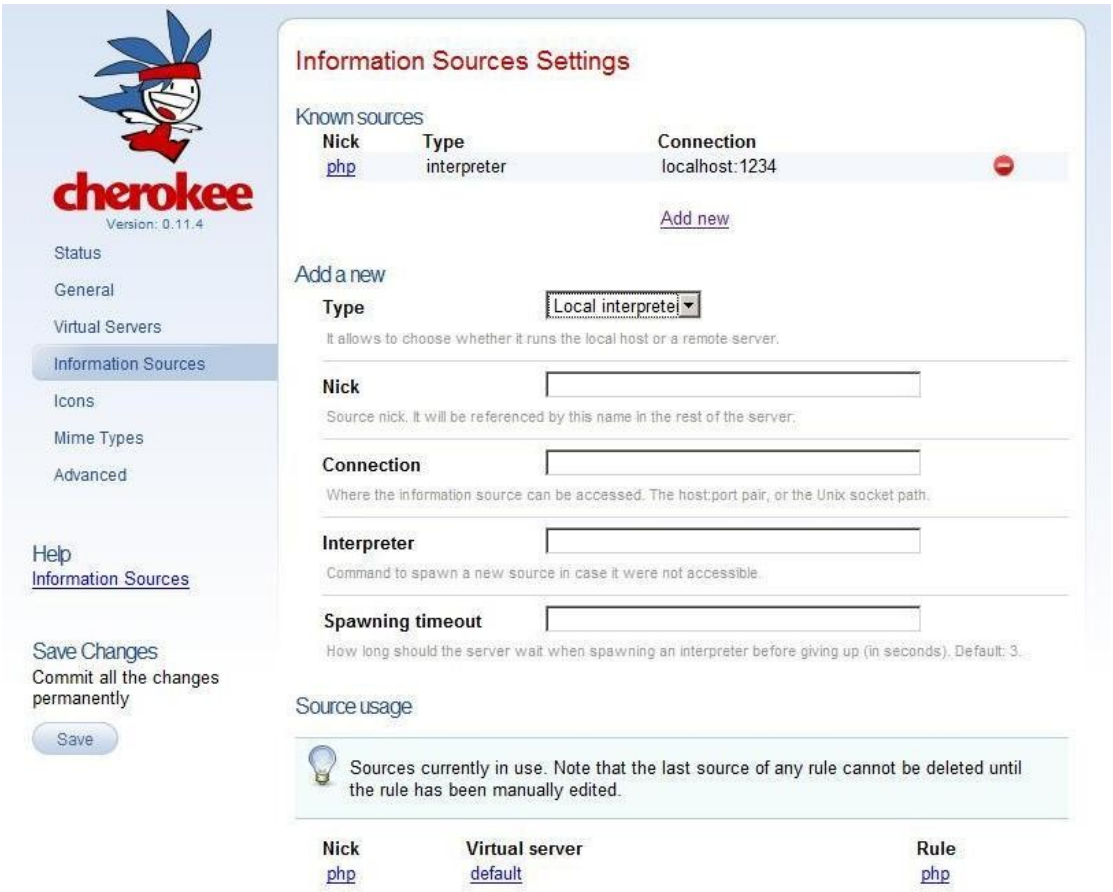
General （常规设置：cherokee 监听的端口、ip 地址、服务器标识显示、Chroot 设置、服务器运行的系统用户等）



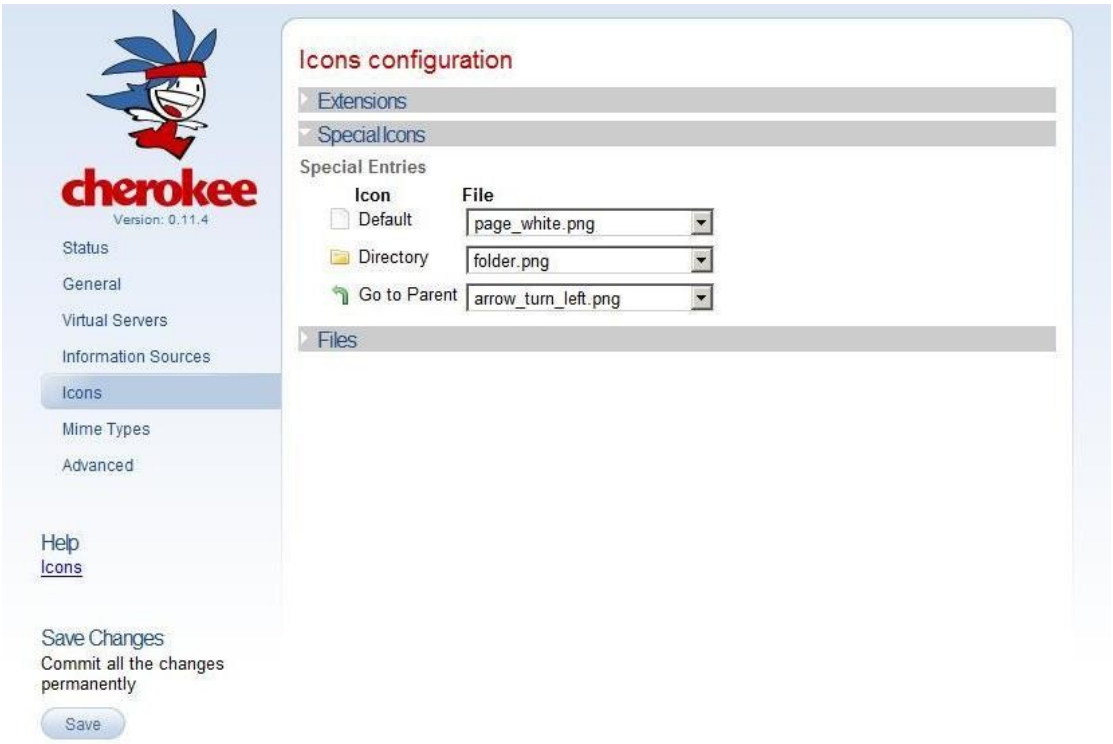
VirtualServers （虚拟主机设置：非常简单，输入主机头以及虚拟主机的文件系统根路径即可，还提供了类似 zeus 的克隆虚拟主机的功能）



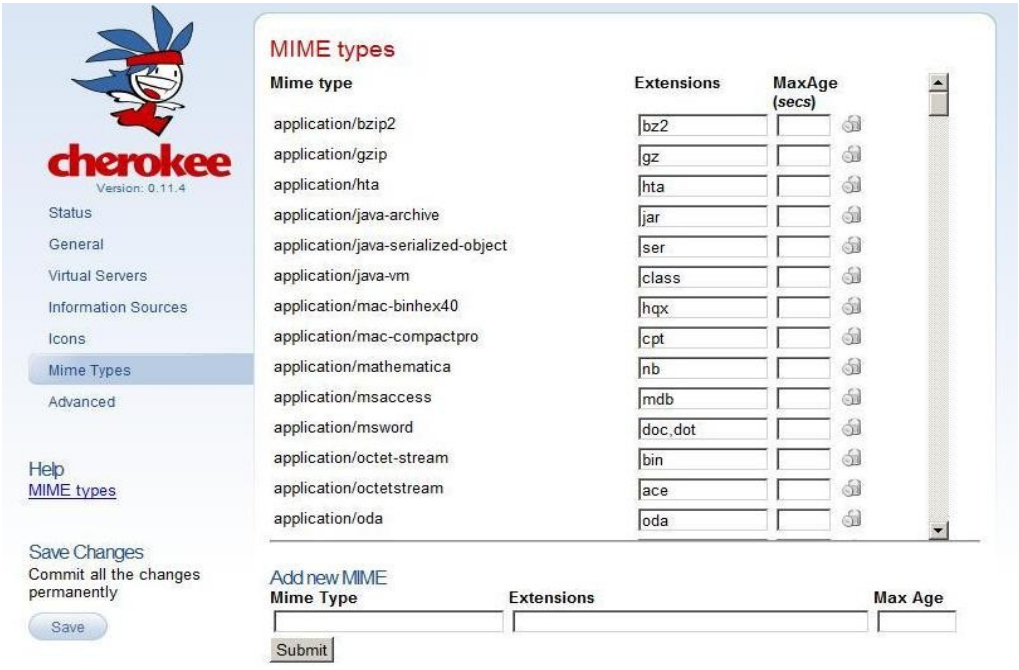
Infomation Sources （信息源：这个可以说是 CGI、FastCGI 设置的地方，你可以添加本地的 fastcgi，也可以添加远程的，很方便）



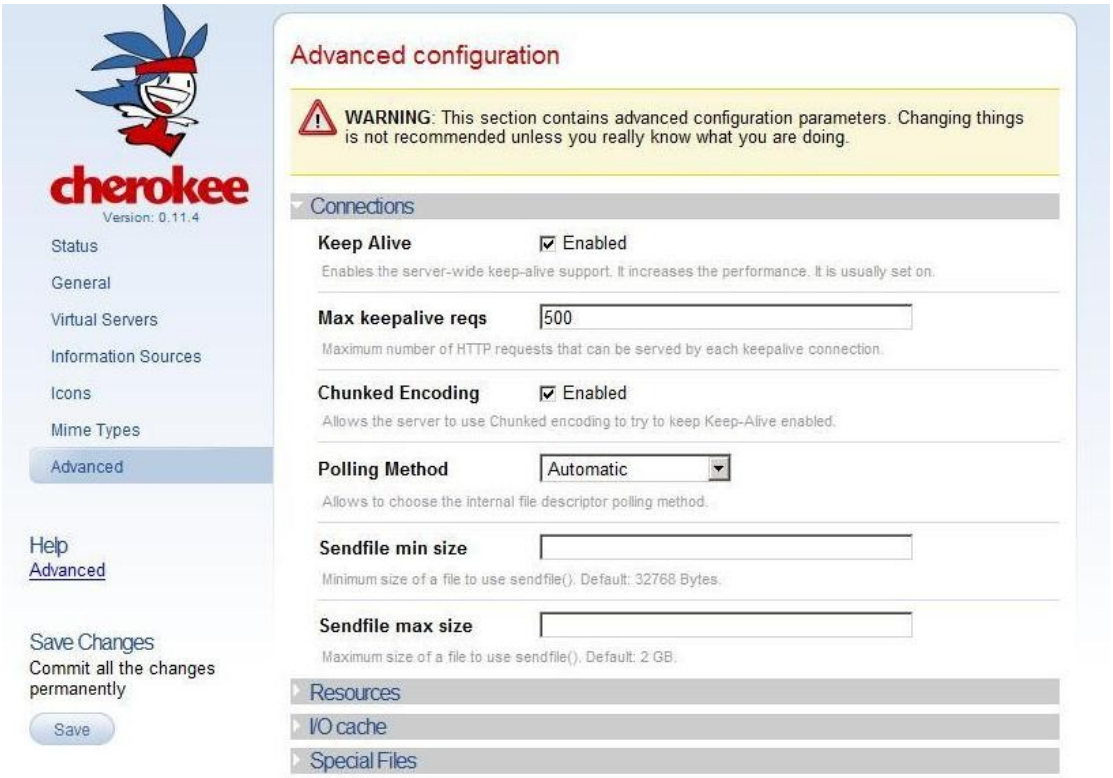
Icons （图标设置：貌似是用来设置在进行目录列表时，对特定的文件类型制定文件图标的）



Mime Types （这个不用说了吧…）



Advanced （高级设置：可以根据你服务器的环境，在这里设置一些 cherokee 比较底层的参数，进一步挖掘 cherokee 的性能）



nagios 监控系统配置过程

ChinaUnix 网友：wiliiwin

这篇文章是从网上看了很多资料，通过自己实践，测试可以完成服务器的基本监控，然后整理而成，希望可以帮助学习 linux 的朋友。

有很多朋友说我的文章和他做的有点出入，我在这里说明下我的 nagios 是 2.9 的，现在最新的是 3.0.5 的，我也测试过了，3.0.5 的和 2.9 的在配置文件上是有点变化的，不要死定着这篇文章，文章只是引导大家去学习原理，配置文件的变动还需要大家去相应的改动。

实验环境：nagios 监控服务器为 192.168.1.240
主机名为 nagios
nagios 被监控服务器为 192.168.1.208
主机名为 apache
用到的软件包：httpd-2.2.6.tar.gz、imagepak-base.tar.gz、mysql-5.1.22-rc-linux- i686-icc-glibc23.tar.gz、nagios-2.9.tar.gz、nagios-plugins-1.4.9.tar.gz、 nrpe-2.12.tar.gz、perl-stable.tar.gz、php-5.2.4.tar.bz2、pnp-latest.tar.gz、 rrdtool-1.0.50.tar.gz

在监控服务器上进行下面的操作：
1.安装 nagios 主程序
1）解压缩：
tar -zxvf nagios-2.9.tar.gz
cd nagios-2.9
./configure --prefix=/usr/local/nagios --with-gd-lib=/usr/local/lib --with-gd-inc=/usr/local/include

2）创建用户并且设定权限：
groupadd nagios
useradd -g nagios nagios
mkdir /usr/local/nagios
chown -R nagios.nagios /usr/local/nagios

make all
make install //来安装主程序,CGI 和 HTML 文件
make install-init //在/etc/rc.d/init.d 安装启动脚本
make install-commandmode //来配置目录权限
make install-config
 //来安装示例配置文件,安装的路径是/usr/local/nagios/etc.

3）验证是否安装成功：
验证程序是否被正确安装。切换目录到安装路径（这里是/usr/local/nagios）,看是否存在 etc、bin、 sbin、 share、 var 这五个目录，如果存在则可以表明程序被正确的安装到系统了。后表是五个目录功能的简要说明：
bin Nagios 执行程序所在目录，nagios 文件即为主程序
etc Nagios 配置文件位置，初始安装完后，只有几个*.cfg-sample 文件
sbin Nagios Cgi 文件所在目录，也就是执行外部命令所需文件所在的目录

share Nagios 网页文件所在的目录
var Nagios 日志文件、spid 等文件所在的目录

var/archives Empty directory for the archived logs
/var/rw Empty directory for the external command file

2.安装插件
1）解压缩：
tar -zxvf nagios-plugins-1.4.9.tar.gz
cd nagios-plugins-1.4.9
./configure --prefix=/usr/local/nagios/
(在 redhat 系统上面安装可能出现 configure 时，到这里 checking for redhat spopen problem...就不动了，所以需要在 configure 时再加上--enable-redhat-pthread-workaround)
make

make install
ls /usr/local/nagios/libexec/
会显示安装的插件文件,即所有的插件都安装在 libexec 这个目录下

注意：要是没有这个插件目录需要用下面的命令把插件复制过来
cp /usr/local/nagios-plugins/libexec /usr/local/nagios/

2)将 apache 的运行用户加到 nagios 组里面：
从 httpd.conf 中过滤出当前的 apache 运行用户
grep ^User /usr/local/apache2/conf/httpd.conf

我的是 daemon，下面将这个用户加入 nagios 组
usermod -G nagios daemon

3)修改 apache 配置:
修改 apache 的配置文件,增加 nagios 的目录,并且访问此目录需要进行身份验证
vi /usr/local/apache2/conf/httpd.conf,在最后增加如下内容：

```
ScriptAlias /nagios/cgi-bin /usr/local/nagios/sbin
<Directory "/usr/local/nagios/sbin">
   Options ExecCGI
   AllowOverride None
   Order allow,deny
   Allow from all
   AuthName "Nagios Access"
   AuthType Basic
AuthUserFile /usr/local/nagios/etc/htpasswd   //用于此目录访问身份验证的文件
   Require valid-user
</Directory>

Alias /nagios /usr/local/nagios/share
<Directory "/usr/local/nagios/share">
   Options None
   AllowOverride None
   Order allow,deny
   Allow from all
   AuthName "Nagios Access"
   AuthType Basic
AuthUserFile /usr/local/nagios/etc/htpasswd  //用于此目录访问身份验证的文件
Require valid-user
</Directory>
```

4)增加验证用户:

也就是通过 web 访问 nagios 的时候,必须要用这个用户登陆.在这里我们增加用户 test:密码为 123456
#/usr/local/apache2/bin/htpasswd -c /usr/local/nagios/etc/htpasswd test //用户名
New password: //(输入 123456)
Re-type new password: (再输入一次密码)
Adding password for user test

5)查看认证文件的内容:
[root@localhost conf]# less /usr/local/nagios/etc/htpasswd
test:OmWGEsBnoGpIc //前半部分是用户名 test,后面是加密后的密码

到这里 nagios 的安装也就基本完成了,你可以通过 web 来访问了.
http://192.168.1.240/nagios 会弹出对话框要求输入用户名密码
输入 test,密码 123456,就可以进入 nagios 的主页面了
但是可以发现什么也点不开,因为 nagios 还没启动呢!下面的工作就是修改配置文件,增加要监控的主机和服务

3.典型配置
nagios 要用起来,就必须修改配置文件,增加要监控的主机和服务才行.在具体做这个动作之前,下面的概念必须要了解.

1)预备知识:
在 Nagios 里面定义了一些基本的对象,一般用到的有:
联系人 contact 出了问题向谁报告?一般当然是系统管理员了
监控时间段 timeperiod 7X24 小时不间断还是周一至周五,或是自定义的其他时间段
被监控主机 Host 所需要监控的服务器,当然可以是监控机自己
监控命令 command nagios 发出的哪个指令来执行某个监控,这也是自己定义的
被监控的服务 Service 例如主机是否存活,80 端口是否开,磁盘使用情况或者自定义的服务等

注意:多个被监控主机可以定义为一个主机组,多个联系人可以被定义为一个联系人组

2)将示例配置文件复制为真实配置文件名:

cd /usr/local/nagios/etc
把这里.cfg-sample 文件配置文件模板,全部重命名为.cfg

3)修改配置文件:
修改 nagios 的主配置文件 nagios.cfg
vi nagios.cfg
cfg_file=/usr/local/nagios/etc/localhost.cfg //在前面加#
cfg_file=/usr/local/nagios/etc/contacts.cfg //联系人配置文件路径
cfg_file=/usr/local/nagios/etc/contactgroups.cfg //联系人组配置文件路径

cfg_file=/usr/local/nagios/etc/commands.cfg　　//命令配置文件路径
cfg_file=/usr/local/nagios/etc/host.cfg　　//主机配置文件路径
cfg_file=/usr/local/nagios/etc/hostgroups.cfg　//服务器组配置文件
cfg_file=/usr/local/nagios/etc/templates.cfg　//模板配置文件路径
cfg_file=/usr/local/nagios/etc/timeperiods.cfg　//监视时段配置文件路径
cfg_file=/usr/local/nagios/etc/services.cfg　　//服务配置文件

其他配置文件以实际情况来进行配置
改 check_external_commands=0 为 check_external_commands=1 .这行的作用是允许在 web 界面下执行重启 nagios、停止主机/服务检查等操作。

把 command_check_interval 的值从默认的 1 改成 command_check_interval=10s（根据自己的情况定这个命令检查时间间隔，不要太长也不要太短）。
主配置文件要改的基本上就是这些，通过上面的修改，发现/usr/local/nagios/etc 并没有文件 hosts.cfg 等一干文件，怎么办？稍后手动创建它们。

4)然后检查配置文件是否出错
/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

出现 Total Warnings: 0
　　Total Errors:　0　　//这样表示配置文件没有错误

5)修改 CGI 脚本控制文件 cgi.cfg

#vi cgi.cfg
第二个要修改的配置文件是 cgi.cfg,它的作用是控制相关 cgi 脚本。先确保 use_authentication=1。接下来修改 default_user_name=test(前面创建的用户名) ,再后面的修改在下表列出：
authorized_for_system_information=nagiosadmin,mandahang，test　　//后面跟的都是用户名
authorized_for_configuration_information=nagiosadmin,mandahang，test
authorized_for_system_commands=mandahang,test
authorized_for_all_services=nagiosadmin,mandahang,test
authorized_for_all_hosts=nagiosadmin,mandahang,test
authorized_for_all_service_commands=nagiosadmin,mandahang,test
authorized_for_all_host_commands=nagiosadmin,mandahang,test

注意：在上面的配置文件里面加上新加的用户 test
那么上述用户名打那里来的呢？是执行命令 /usr/local/apache2/bin/htpasswd -c /usr/local/nagios/etc/htpasswd test（用户名 test） 所生成的

6)配置各种配置文件

定义监控时间段,创建配置文件 timeperiods.cfg :

```
# vi timeperiods.cfg
define timeperiod{
        timeperiod_name         24x7   //时间段的名称,这个地方不要有空格
        alias                   24 Hours A Day,7Days A Week
        sunday                  00:00-24:00
        monday                  00:00-24:00
        tuesday                 00:00-24:00
        wednesday               00:00-24:00
        thursday                00:00-24:00
        friday                  00:00-24:00
        saturday                00:00-24:00
        }
```

定义了一个监控时间段,它的名称是 24x7,监控的时间是每天全天 24 小时:

定义联系人,创建配置文件 contacts.cfg

```
# vi contacts.cfg
define contact{
        contact_name                    test    //联系人的名称,这个地方不要有空格
        alias                           sys admin
        service_notification_period     24x7
        host_notification_period        24x7
        service_notification_options    w,u,c,r
        host_notification_options       d,u,r
        service_notification_commands   notify-by-email
        host_notification_commands      host-notify-by-email
email                           tfhudong@sohu.com
        pager                           1391119xxxx
        }
```

创建了一个名为 test 的联系人,下面列出其中重要的几个选项做说明:

service_notification_period 24x7
服务出了状况通知的时间段,这个时间段就是上面在 timeperiods.cfg 中定义的.

host_notification_period 24x7
主机出了状况通知的时间段, 这个时间段就是上面在 timeperiods.cfg 中定义的

service_notification_options w,u,c,r
当服务出现 w—报警(warning),u—未知(unkown),c—严重(critical),或者 r—从异常情况恢复正常,在这四种情况下通知联系人.

host_notification_options d,u,r
当主机出现 d—当机(down),u—返回不可达(unreachable),r—从异常情况恢复正常,在这 3 种情况下通知联系人
service_notification_commands notify-by-email
服务出问题通知采用的命令 notify-by-email,这个命令是在 commands.cfg 中定义的,作用是给联系人发邮件.

host_notification_commands host-notify-by-email
同上,主机出问题时采用的也是发邮件的方式通知联系人

email yahoon@test.com
很明显,联系的人 email 地址
Pager 137xxxxxxxxxxxxx //电话
}

下面就可以将多个联系人组成一个联系人组,创建文件 contactgroups.cfg :

```
# vi contactgroups.cfg
define contactgroup{
      contactgroup_name      sagroup    //联系人组的名称,同样不能空格
    alias                    System Administrators       //别名
    members                    test
//组的成员,来自于上面定义的 contacts.cfg,如果有多个联系人则以逗号相隔
      }
```

定义被监控主机,创建文件 hosts.cfg :

```
# vi hosts.cfg
define host{
      host_name       nagios       //被监控主机的名称,最好别带空格
      alias          nagios          //别名
     address    192.168.1.240   //被监控主机的 IP 地址
     check_command    check-host-alive
      //监控的命令 check-host-alive,这个命令来自 commands.cfg,用来监控主机是否存活
      max_check_attempts    5  //检查失败后重试的次数
      check_period         24x7
    //检查的时间段 24x7,同样来自于我们之前在 timeperiods.cfg 中定义的
contact_groups       sagroup
    //联系人组,上面在 contactgroups.cfg 中定义的 sagroup
notification_interval  10
     //提醒的间隔,每隔 10 秒提醒一次
notification_period   24x7
     //提醒的周期, 24x7,同样来自于我们之前在 timeperiods.cfg 中定义的
notification_options          d,u,r
```

```
//指定什么情况下提醒
    }
```

通过简单的复制修改就可以定义多个主机了.我们在这加上另外一台机器

与联系人可以组成联系人组一样,多个主机也可以组成主机组:
主机名为：apache　　ip：192.168.1.208

创建文件 hostgrops.cfg

```
# vi hostgroups.cfg
define hostgroup{
    hostgroup_name      linux-servers  //主机组名称
    alias               linux-servers  //别名
    members             nagios,apache
//组的成员主机,多个主机以逗号相隔,必须是上面 hosts.cfg 中定义的
    }
```

下面是最关键的了,用 nagios 主要是监控一台主机的各种信息,包括本机资源,对外的服务等等.这些在 nagios 里面都是被定义为一个个的项目 (nagios 称之为服务,为了与主机提供的服务相区别,我这里用项目这个词),而实现每个监控项目,则需要通过 commands.cfg 文件中定义的命 令.
例如我们现在有一个监控项目是监控一台机器的 web 服务是否正常, 我们需要哪些元素呢?最重要的有下面三点:首先是监控哪台机,然后是这个监控要用什么命令实现,最后就是出了问题的时候要通知哪个联系人?

定义监控的项目,也叫服务,创建 services.cfg :

```
# vi services.cfg
define service{
    host_name           nagios
    //被监控的主机,hosts.cfg 中定义的
    service_description     check-host-alive
    //这个监控项目的描述,这个会在 web 页面中出现
    check_command       check-host-alive
    //所用的命令,是 commands.cfg 中定义的
    max_check_attempts      5   //重试的次数
    normal_check_interval   3   //循环检查的间隔时间
    retry_check_interval    2
    check_period        24x7
    //监控的时间段,是 timeperiods.cfg 中定义的
    notification_interval   10
    notification_period     24x7
    //通知的时间段
notification_options    w,u,c,r
```

```
//在监控的结果是 wucr 时通知联系人
contact_groups        sagroup
//联系人组,是 contactgroups.cfg 中定义的
}
```

这样整个的配置过程就结束了.虽然功能很简单,但是已经为以后扩展打下了良好的基础。在运行 nagios 之前首先做测试：

/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

看到下面这些信息就说明没问题了

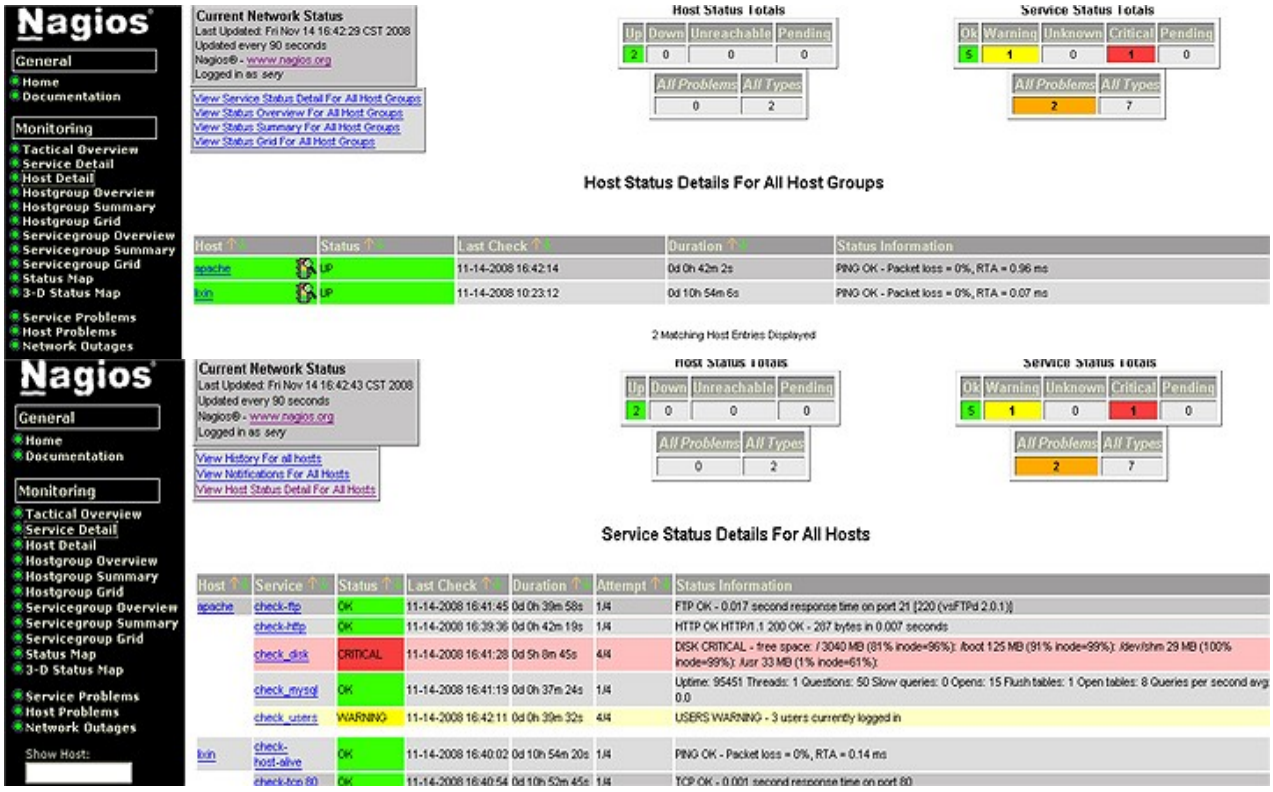
Total Warnings: 0

Total Errors: 0

作为守护进程后台启动 nagios

/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

登陆 http://192.168.1.240/nagios/来查看吧.点左边的 Host Detail



4、使用命令和插件监控更多信息

我们已经增加了二个监控项目,分别监控 nagios,apache 这二台主机是否存活。nagios 本身并没有监控的功能，所有的监控是由插件完成的。插件将监控的结果返回给 nagios,nagios 分析这些结果,以 web 的方式展现给我们，同时提供相应的报警功能(这个报警的功能也是由插件完成的) 所有的这些插件是一些实现特定功能的可执行程序,默认安装的路径是/usr/local/nagios/libexec。

这些程序都是可以独立执行的,使用方法可以通过”命令名 –h”来查看，例如,我们查看

check_disk 这个插件的用法则可以使用 check_disk –h
./check_disk -h
check_disk (nagios-plugins 1.4.9) 1.91
Copyright (c) 1999 Ethan Galstad <nagios@nagios.org>
Copyright (c) 1999-2006 Nagios Plugin Development Team
<[email]nagiosplug-devel@lists.sourceforge.net[/email]>
This plugin checks the amount of used disk space on a mounted file system
and generates an alert if free space is less than one of the threshold values
Usage: check_disk -w limit -c limit [-p path | -x device] [-t timeout][-m] [-e] [-W limit] [-K limit] [-v] [-q] [-E]

输出的资料十分详细给出了这个插件的功能,使用方法,参数意义等,对于每一个插件都是这样.所以当你不懂某个插件怎么使用时就好好读读吧.从上面的输出可以看到 check_disk 这个插件是用来检查磁盘使用情况的.
我现在来独立执行它,例如查看根分区的使用情况,执行
./check_disk -w 10% -c 5% /
命令的含义是检查分区/的使用情况,若剩余 10%以下,为警告状态(warning),5%以下为严重状态(critical),

执行后我们会看到下面这条信息
DISK WARNING - free space: / 487 MB (6% inode=78%); | /=7449MB;7524;7942;0;8361
说明当前是 warning 的状态,空闲空间只有 6%了.如果 nagios 收到这些状态结果就会采取报警等措施了

我们在定义某个监控项目时,所用的监控命令都是来自 commands.cfg 的,这和这些插件有什么关系?想到了吧,commands.cfg 中定义的监 控命令就是使用的这些插件.举个例子,之前我们已经不止一次用到了 check-host-alive 这个命令,打开 commands.cfg 就可以看到这 个命令的定义,如下:
define command{
command_name check-host-alive
command_line $USER1$/check_ping -H $HOSTADDRESS$ -w 3000.0,80% -c 5000.0,100% -p 1
}

command_name check-host-alive
这句话的意思是定义的命令名是 check-host-alive,也就是我们在 services.cfg 中使用的名称
执行的操作是
$USER1$/check_ping -H $HOSTADDRESS$ -w 3000.0,80% -c 5000.0,100% -p 1
其中$USER1$是在 resource.cfg 文件中定义的,代表插件的安装路径.就如我们上面看到的那样 $USER1$=/usr/local/nagios/libexec,至于$HOSTADDRESS$,则默认被定义为监控主机的地址.
简单的说,我们在 services.cfg 中定义了对 dbpi 执行 check-host-alive 命令,实际上就是执行了 /usr/local/nagios/libexec/ check_ping -H dbpi 的 ip 地址 -w 3000.0,80% -c 5000.0,100% -p 1
实际上 check-host-alive 只是这一长串命令的简称而已,而在 services.cfg 中都是使用简称的.
在 commands.cfg 中定义了很多这样的命令简称.基本上我们常用的监控项目都包含了,例如

ftp,http,本地的磁盘,负载等等.
我们再看一个命令,check_local_disk 定义如下
define command{
command_name check_local_disk
command_line $USER1$/check_disk -w $ARG1$ -c $ARG2$ -p $ARG3$
}

check_local_disk 实际上是执行的 check_disk 插件.这里的$ARG1$, $ARG2$, $ARG3$是什么意思呢?在之前我们已经提到了这个 check_disk 这个插件的用法,-w 的参数指定磁盘剩了多少是警告状态,-c 的参数指定剩多少 是严重状态,-p 用来指定路径.
在使用 check-host-alive 的时候,只需要在 services.cfg 中直接写上这个命令名 check-host-alive.后面没任何的参数.而使用 check_local_disk 则不同,在 services.cfg 中这要这么写
check_local_disk!10%!5%!/
在命令名后面用!分隔出了 3 个参数,10%是$ARG1$的值,5%是$ARG2$的值,/ 是$ARG3$的值,

简单的一句话就是
services.cfg 定义监控项目用某个命令
↓
这个命令必须在 commands.cfg 中定义
↓
定义这个命令时使用了 libexec 下的插件

如果命令不带$ARG1$就可以在 services.cfg 中直接使用,如果带了使用时就带上参数,以!相隔

1).监控 nagios 的 ftp
编辑 services.cfg 增加下面的内容,基本上就是 copy 上节我们定义监控主机存活的代码.
define service{
host_name nagios
//要监控的机器,给出机器名,注意必须是 hosts.cfg 中定义的
service_description check ftp
//给这个监控项目起个名字吧,任意起,你自己懂就行
check_command check_ftp
max_check_attempts 5
normal_check_interval 3
retry_check_interval 2
check_period 24x7
notification_interval 10
notification_period 24x7
notification_options w,u,c,r
contact_groups sagroup
}

修改了配置文件,当然就要重新启动了,简单的方法杀掉 nagios 进程,然后重新启动

/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

这里只能监控监控服务器的本地信息，需要监控被监控服务器的信息，就需要下面的操作。

5.用 NRPE 监控 LINUX 上的”本地信息”
本节的目的,对系统为 linux 的主机 apcahe 进行如下监控:磁盘容量,登陆用户数，http,ftp,mysql 的状况,其他功能自己可以自定义配置

对于像磁盘容量,cpu 负载这样的”本地信息”,nagios 只能监测自己所在的主机,而对其他的机器则显得有点无能为力.毕竟没得到被控主机的适当权限 是不可能得到这些信息的.为了解决这个问题,nagios 有这样一个附加组件----NRPE.用它就可以完成对 linux 类型主机”本地信息”的监控.

所以我们按照图示在监控主机(nagios)和被监控主机上安装相应的软件

在被监控主机上

1)增加用户
useradd nagios

设置密码
passwd nagios

2)安装 nagios 插件
tar -zxvf nagios-plugins-1.4.9.tar.gz
cd nagios-plugins-1.4.9
编译安装
./configure --enable-redhat-pthread-workaround
make
make install
这一步完成后会在/usr/local/nagios/下生成两个目录 libexec 和 share
ls /usr/local/nagios/
libexec share

修改目录权限
chown nagios.nagios /usr/local/nagios
chown -R nagios.nagios /usr/local/nagios/libexec

3)安装 nrpe
解压缩
tar -zxvf nrpe-2.8.1.tar.gz
cd nrpe-2.8.1
编译
./configure

NRPE 的端口是 5666
make all
接下来安装 NPRE 插件,daemon 和示例配置文件

安装 check_nrpe 这个插件
make install-plugin
之前说过监控机需要安装 check_nrpe 这个插件,被监控机并不需要,我们在这里安装它是为了测试的目的

安装 deamon
make install-daemon

安装配置文件
make install-daemon-config

现在再查看 nagios 目录就会发现有 4 个目录了
ls /usr/local/nagios/
bin etc libexec share
将 NRPE deamon 作为 xinetd 下的一个服务运行的.在这样的情况下 xinetd 就必须要先安装好,不过一般系统已经默认装了

4)安装 xinetd 脚本
make install-xinetd
输出如下
/usr/bin/install -c -m 644 sample-config/nrpe.xinetd /etc/xinetd.d/nrpe

可以看到创建了这个文件/etc/xinetd.d/nrpe
编辑这个脚本
vi /etc/xinetd.d/nrpe
```
# default: on
# description: NRPE (Nagios Remote Plugin Executor)
service nrpe
{
        flags           = REUSE
        socket_type     = stream
        port            = 5666
        wait            = no
        user            = nagios
        group           = nagios
        server          = /usr/local/nagios/bin/nrpe
        server_args     = -c /usr/local/nagios/etc/nrpe.cfg --inetd
        log_on_failure  += USERID
        disable         = no
```

```
        only_from       = 127.0.0.1    在后面增加监控主机的地址 192.168.1.240,以空格间隔
}
```

改后

```
    only_from       = 127.0.0.1 192.168.1.240
```

编辑/etc/services 文件,增加 NRPE 服务

```
vi /etc/services
```

增加如下

```
# Local services
nrpe            5666/tcp                    # nrpe
```

重启 xinetd 服务

```
# service xinetd restart
```

查看 NRPE 是否已经启动

```
# netstat -at | grep nrpe
tcp        0      0 *:nrpe                  *:*                     LISTEN
# netstat -an | grep 5666
tcp        0      0 0.0.0.0:5666            0.0.0.0:*               LISTEN
```

可以看到 5666 端口已经在监听了

5)测试 NRPE 是否则正常工作

之前安装了 check_nrpe 这个插件用于测试,现在就是用的时候.执行

```
/usr/local/nagios/libexec/check_nrpe -H localhost
```

会返回当前 NRPE 的版本

```
# /usr/local/nagios/libexec/check_nrpe -H localhost
NRPE v2.8.1
```

也就是在本地用 check_nrpe 连接 nrpe daemon 是正常的

注:为了后面工作的顺利进行,注意本地防火墙要打开 5666 能让外部的监控机访问

/usr/local/nagios/libexec/check_nrpe –h 查看这个命令的用法

可以看到用法是 check_nrpe –H 被监控的主机 -c 要执行的监控命令

注意:-c 后面接的监控命令必须是 nrpe.cfg 文件中定义的.也就是 NRPE daemon 只运行 nrpe.cfg 中所定义的命令

查看 NRPE 的监控命令

```
cd /usr/local/nagios/etc
vi nrpe.cfg
```

```
# The following examples use hardcoded command arguments...
command[check_users]=/usr/local/nagios/libexec/check_users -w 5 -c 10
command[check_load]=/usr/local/nagios/libexec/check_load -w 15,10,5 -c 30,25,20
command[check_hda1]=/usr/local/nagios/libexec/check_disk -w 20 -c 10 -p /dev/hda1
command[check_zombie_procs]=/usr/local/nagios/libexec/check_procs -w 5 -c 10 -s Z
command[check_total_procs]=/usr/local/nagios/libexec/check_procs -w 150 -c 200
```

注意：其他命令需要自行添加

也就是 check_nrpe 的-c 参数可以接的内容,等号=后面是实际执行的插件程序(只这与 commands.cfg 中定义命令的形式十分相似,不过是写在了一行).也就是说 check_users 就是等号后面/usr/local/nagios/libexec/check_users -w 5 -c 10 的简称.
我们可以很容易知道上面这 5 行定义的命令分别是检测登陆用户数,cpu 负载,hda1 的容量,僵尸进程,总进程数.各条命令具体的含义见插件用法(执行”插件程序名 –h”)
由于-c 后面只能接 nrpe.cfg 中定义的命令,也就是说现在我们只能用上面定义的这五条命令.我们可以在本机实验一下.执行
/usr/local/nagios/libexec/check_nrpe -H localhost -c check_users

/usr/local/nagios/libexec/check_nrpe -H localhost -c check_load

/usr/local/nagios/libexec/check_nrpe -H localhost -c check_hda1

/usr/local/nagios/libexec/check_nrpe -H localhost -c check_zombie_procs

/usr/local/nagios/libexec/check_nrpe -H localhost -c check_total_procs

在运行 nagios 的监控主机上

之前已经将 nagios 运行起来了,现在要做的事情是:
- 安装 check_nrpe 插件
- 在 commands.cfg 中创建 check_nrpe 的命令定义,因为只有在 commands.cfg 中定义过的命令才能在 services.cfg 中使用

创建对被监控主机的监控项目

安装 check_nrpe 插件

```
# tar -zxvf nrpe-2.8.1.tar.gz
# cd nrpe-2.8.1
# ./configure
# make all
# make install-plugin
```

只运行这一步就行了,因为只需要 check_nrpe 插件

在 apache 刚装好了 nrpe,现在我们测试一下监控机使用 check_nrpe 与被监控机运行的 nrpedaemon 之间的通信.

/usr/local/nagios/libexec/check_nrpe -H 192.168.1.208
NRPE v2.8.1

看到已经正确返回了 NRPE 的版本信息,说明一切正常.

在 commands.cfg 中增加对 check_nrpe 的定义
vi /usr/local/nagios/etc/commands.cfg
在最后面增加如下内容
##
#
2007.9.5 add by yahoon
NRPE COMMAND
#
##
'check_nrpe ' command definition
define command{
 command_name check_nrpe
 command_line $USER1$/check_nrpe -H $HOSTADDRESS$ -c $ARG1$
 }

意义如下
command_name check_nrpe
定义命令名称为 check_nrpe,在 services.cfg 中要使用这个名称.

command_line $USER1$/check_nrpe -H $HOSTADDRESS$ -c $ARG1$
这是定义实际运行的插件程序.这个命令行的书写要完全按照 check_nrpe 这个命令的用法.不知道用法的就用 check_nrpe –h 查看

-c 后面带的$ARG1$参数是传给 nrpe daemon 执行的检测命令,之前说过了它必须是 nrpe.cfg 中所定义的那 5 条命令中的其中一条.在 services.cfg 中使用 check_nrpe 的时候要用!带上这个参数

下面就可以在 services.cfg 中定义对 apache 主机磁盘容量的监控
define service{
 host_name apache
被监控的主机名,这里注意必须是 linux 且运行着 nrpe,而且必须是 hosts.cfg 中定义的
 service_description check-disk
 监控项目的名称

check_command check_nrpe!check_disk
监控命令是 check_nrpe,是在 commands.cfg 中定义的,带的参数是 check_disk,是在 nrpe.cfg 中定义的
max_check_attempts 5
normal_check_interval 3
retry_check_interval 2
check_period 24x7
notification_interval 10
notification_period 24x7
notification_options w,u,c,r
contact_groups sagroup
}
像这样将其余几个监控项目加进来.

6.Nagios 的性能分析图表(此安装在监控机上进行)

Nagios 监控的侧重点在“此时”服务是否正常，是一个瞬时状态。通过对这个状态的监控和告警，管理员可以第一时间对主机或者服务的故障做处理。但是我 们往往也非常关心主机的性能以及服务的响应时间等情况，这些情况是一个持续的变化曲线，并非一个实时的值，如果通过查看日志数据来分析的话，既繁琐有抽 象，所以，我们希望 Nagios 可以帮我们做这份工作，然后将报表提交给我们，这样就非常方便了。这就是今天需要用到的 Nagios 的相关开源项目—— PNP

PNP 是一个小巧的开源软件包，它是基于 PHP 和 PERL，利用 rrdtool 将 Nagios 采集的数据绘制成图表。如果你要安装 PNP，那么准备工作有如下 3 项：
1、整合 Apache 和 PHP
2、安装 rrdtools
3、安装 Perl
1）安装 php

```
./configure \
--prefix=/usr/local/php \
--with-mysql=/usr/local/mysql \
--with-apxs2=/usr/local/apache2/bin/apxs \
--with-gd --with-jpeg-dir=/usr/lib --enable-gd-native-ttf \
--with-zlib-dir=/usr/lib --with-png-dir=/usr/lib \
--with-freetype-dir=/usr/include/freetype2 --with-ttf \
--enable-sockets --enable-ftp --enable-mbstring

make && make install

#在 httpd 配置文件里加入，使 apache 支持 php
AddType application/x-httpd-php .php .phtml
AddType application/x-httpd-php-source .phps

#拷贝 php 配置文件到指定位置
```

```
cp php.ini-dist /usr/local/php/lib/php.ini
```

2）安装 rrdtools

```
./configure
make&&make install
```

3) 安装 Perl

```
rm -f config.sh Policy.sh
sh Configure -de
make
make test
make install
```

要介绍 PNP 工作原理，首先要说明一下 Nagios 提供的数据接口，也就是 PNP 的数据来源。在定义 host 或 service 中都有一个定义项，名为 process_perf_data，其值可以定义为 0 或 1，其作用是是否启用 Nagios 的数据输出功能。如果你将此项赋值为 1，那么 Nagios 就会将收集的数据写入到某个文件中，以备提取。所以，如果你想让 Nagios 将数据输出的话，首先要将 Nagios 的主配置文件 nagios.cfg 中相关的配置修改：

```
process_performance_data=1
service_perfdata_command=process-service-perfdata  #默认此句被注释掉了，在这里去掉前面的#
```

如果想要对某个监控对象做数据图表，则需在所对应的 host 或者 service 定义中（一般写在 hosts.cfg 或者 services.cfg 文件中），包含如下的定义：

```
process_perf_data   1
```

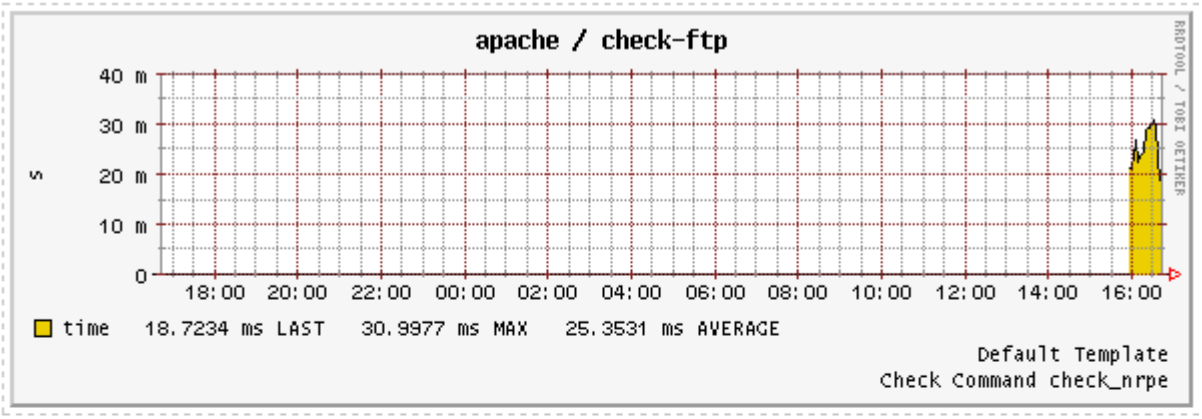
这 样，Nagios 就会调用相应的命令来输出数据了。Nagios 的 command.cfg 定义中默认有一项“process-service- perfdata”，该命令声明了 Nagios 输出哪些值到输出的文件中。不过其定义相对简单，PNP 提供了一个 perl 脚本，更详尽的定义了一个输出数 据的方法。如果要使用 PNP 的话，我们需要在 command.cfg 的定义中，将“process-service-perfdata”命令对应的执行命令行的内 容替换成该脚本：

```
define command{
command_name process-service-perfdata
command_line /usr/local/nagios/libexec/process_perfdata.pl
}
```

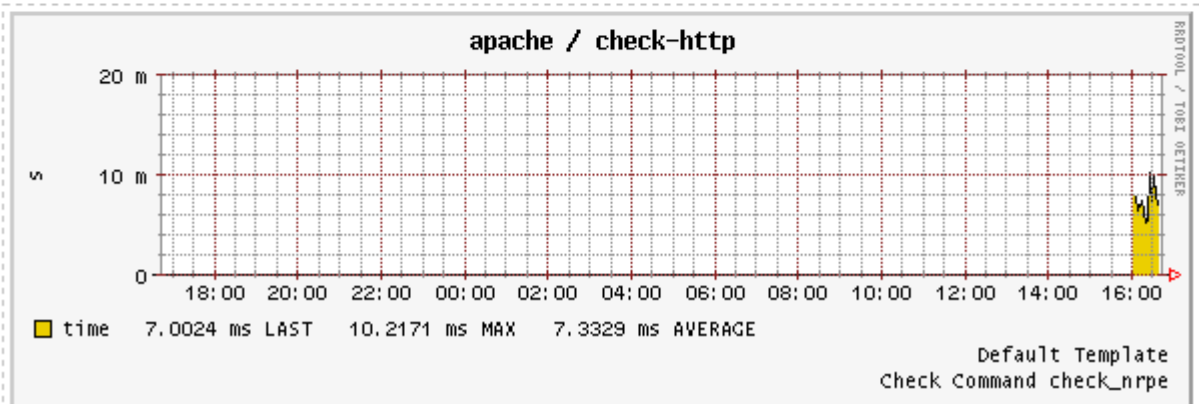
这样设置了之后，Nagios 就会利用 PNP 提供的脚本进行相关的工作了。刚刚定义命令时用到的脚本“process_perfdata.pl”现在还不存在。我们现在就来安装，也就 PNP 的软件包 安装方法很简单，过程也很顺利

./configure --with-rrdtool=/usr/local/rrdtool-1.0.50/bin/rrdtool --with-perfdata-dir=/usr/local/nagios/share/perfdata/
make all
make install

安装结束之后，再去检查一下 Nagios 的插件目录（libexec），就会发现多了一个名为“process_perfdata.pl”的脚本。
ok，现在执行一下 Nagios 配置检查命令
/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
如果没有什么错误，那么我们重新启动 Nagios。
/etc/init.d/nagios restart
验收一下成果，在浏览器的地址栏中输入：
http://IP/nagios/pnp/index.php



24 Hours (13.11.08 16:41 - 14.11.08 16:41)
Service: check-http
Datasource: time



基本上 nagios 的主要功能就有这些,nagios 的使用关键在于如何活用那些丰富的插件.nagios 可以说是一个对于 linux/unix 环境支持十分好的程序.

之前我说重启 nagios 的时候都是用的杀进程的方式,其实也可以不这么做.如果在安装 nagios 的时候安装了启动脚本就可以使用/etc/init.d/nagios restart 还可以带的参数有 stop, start,status
如果报错了,有可能是脚本里面的路径设置错误,解决办法：

vi /etc/init.d/nagios
将 prefix=/usr/local/nagiosaa 改为安装的目录/etc/init.d/nagios

注:在 nagios 安装的时候说是将脚本安装到了/etc/rc.d/init.d,其实这和/etc/init.d 是一个目录。

使用 SystemImager 快速部署系统

ChinaUnix 网友：changzi100

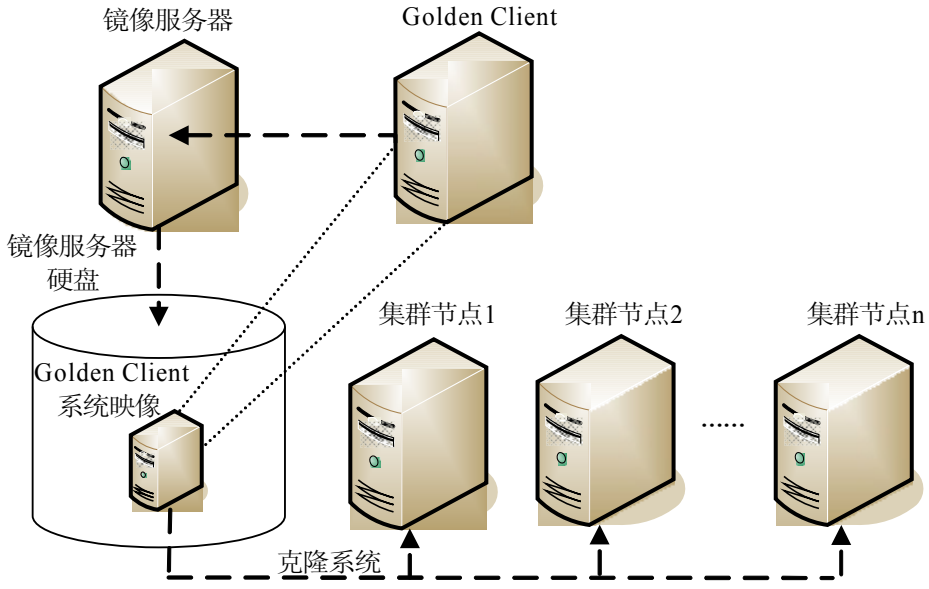
系统部署是构建集群的第一步，如何快速，高效地部署系统是值得讨论的问题。本文描述基于开源软件 SystemImager 的系统部署方法并给出基本操作步骤。

SystemImager 由 Brian Elliott Finley 创建。目前的维护者和项目小组的领导者是 Andrea Righi。SystemImager，是 System Installation Suite 的一部份，它能自动安装 GNU/Linux，发布软件及快速部署生产环境。

SystemImager 的一个主要特征是与发行版本无关，并且能支持不同种类的硬件。这一特征可以让我们将任何种类的 GNU/Linux（标准的发行版本或者自定义的版本）部署到目标机。此项目的主要目标是实现轻松，快捷的部署大量的节点。典型的应用环境包括计算机实验室，render farms（注：视觉农场，很多大片在后期制作所使用的图形渲染的工作站，类似于超级计算机），Internet 服务器中心，数据库服务器中心，公司桌面环境等。SystemImager 已经被证实了在集群计算环境中很流行，如网格计算和高性能计算。

另一个设计上便于 GNU/Linux 发行以及不依赖于硬件的特点是 SystemImager 处理的是基于系统镜像的文件。镜像以文件形式储存目录体系节构，即为样本节点的一个全面的快照，包含节点根文件系统中所有文件及目录。镜像以多种方法获得，包括从目标系统（golden client）中获得，或应用第三方工具直接在镜像服务器中生成。

SystemImage 的工作原理：安装并配置好镜像服务器及样本节点（Golden Client），然后镜像服务器捕捉样本节点的镜像并存储，最后启动目标节点并将镜像分发下去，完成对目标节点的部署。如下图所示：



具体步骤：

前一阵在我的博客中针对低版本（SystemImager3.2）的安装及使用写了一些步骤，现在针对现在的稳定版本 4 写一下具体操作步骤。

实验环境：服务器 P4-1.7/256M，CentOS5.1；在另一台 P4-2.8/1G 的机器上安装几台虚拟机作为节点，虚拟机分配 256M 内存，其中一台作为 Golden Client，所装系统为 CentOS5.2，大小约 1.6G；100Mbits 局域网。

1，安装软件

从 https://sourceforge.net/project/platformdownload.php?group_id=259 下载需要的软件版本和相应硬件架构的启动包。SystemImage 用 Perl 开发，所以安装时对 Perl 模块有依赖性。如 AppConfig，MLDBM，XML-Simple 等，需要时可以自己下载安装，此处不做详细说明。

镜像服务器中执行如下命令：

```
# rpm -ivh --nodeps systemconfigurator-2.2.11-1.noarch.rpm \
> systemimager-server-4.0.2-1.noarch.rpm \
> systemimager-common-4.0.2-1.noarch.rpm \
> systemimager-i386boot-standard-4.0.2-1.noarch.rpm \
> systemimager-i386initrd_template-4.0.2-1.noarch.rpm
```

Golden Client 中执行如下命令：

```
# rpm -ivh --nodeps systemconfigurator-2.2.11-1.noarch.rpm \
> systemimager-common-4.0.2-1.noarch.rpm \
> systemimager-client-4.0.2-1.noarch.rpm \
> systemimager-i386initrd_template-4.0.2-1.noarch.rpm
```

镜像服务器及 Golden Client 中已经安装所需的 Perl 模块，所以加上--nodeps 参数。如果此处没安装有的话系统会提示依赖性，可根据提示下载并安装相应软件包。

2，Golden Client 端准备

在 golden client 上，以 root 执行 si_prepareclient 命令。这会在/etc/systemimager 目录下生成许多文件，其中包含分区方案，文件系统类型等。si_prepareclient 会启动一个 rsync 进程来让这些文件传输到服务器上。

```
# si_prepareclient --server 192.168.1.63
```

服务器地址为 192.168.1.63，此处也可以写服务器名称。

执行完上面的命令后系统会以命令行交互方式询问是否继续，其中会更改一些文件配置并启动 rsync 进程，如果回答 y 继续的话，系统会告知已经准备完毕，可以在服务器端执行 si_getimage 命令。

3，服务器端获取镜像

在镜像服务器上执行 si_getimage，从 golden client 捕捉镜像。应用 si_getimage，镜像服

务器从样本节点文件系统的根目录中将所有文件及目录备份成镜像存放于/var/lib/systemimager/images 中。

si_getimage --golden-client 192.168.1.67 --image backup

此处要指明 golden client 的地址或主机名以及生成镜像的名称。在询问是否继续后系统会做出反映。如果同意继续并无其它阻碍（防火墙）的话此时会复制 golden client 中的文件。笔者所装大小约为 1.6G 的 golden client 系统（在 VM 中安装的 CentOS5.2）来说复制过程大约 10 分钟，如果是实机的话此处会表现出更好的性能。

4，准备分发镜像

镜像获得后存放于/var/lib/systemimager/images/backup，其中所包含的内容即为 golden client 的系统文件及目录。

可以通过四种方式启动节点来分发镜像：

- 从网络启动（PXE）
- 从自动安装 CD 启动
- 从自动安装盘启动（USB 设备或软盘）
- 从一个启动的系统启动

此处可按不同的硬件环境来选择不同的方法，如果网卡不支持 PXE 的话可以选择用 USB 设备或 CD 进行启动后安装，当然不同的安装方法使用不同的命令创建启动工具。本文介绍了以 PXE 方式启动安装，这种方法的优点是简约而时尚，当然，如果硬件并不支持 PXE 的话可以创建另外的启动工具，对安装的节点来说只是通过不同的方式获得引导信息。

SystemImager 的 si_mkbootserver 命令用来配置启动服务器。执行 si_mkbootserver 启动一个交互过程，它会创建 tftproot 目录，配置 tftp 服务器，并执行一些测试看看功能是否正常。一旦 si_mkbootserver 检测出错误，它会宣告失败并生成错误日志。更正错误后，可以重新执行 si_mkbootserver，不断重复这个过程直到配置成功。si_mkbootserver 同时会调用 si_mkdhcpserver 命令，用来为节点分配 IP。这个命令简化了 DHCP 的配置， 它会询问需要的所有信息来创建适合 SystemImager 安装的 DHCP 配置文件。

下面是笔者进行的配置：

si_mkbootserver

WARNING: this script may modify the following files:

--> /etc/services

--> /etc/inetd.conf

--> /etc/xinetd.d/tftp

And can restart inetd, xinetd or tftp servers.

Do you wish to continue (y/[n])? y

Ok, continuing...

/var/lib/tftpboot exists and is a symlink to /usr/share/systemimager/boot.

Checking for a tftp server... found.

Checking if tftp server is H. Peter Anvin's tftp server... yup - right on!

Checking for a running inetd... Not found.

Checking for a running xinetd... 3262.

Looking for update-inetd... not found.

Backing up /etc/xinetd.d/tftp...

Moving /etc/xinetd.d/tftp to /etc/xinetd.d/tftp.si_mkbootserver.bak1...done.

Restaring xinetd ...

停止 xinetd: [确定]

启动 xinetd: [确定]

done.

Looking for a tftp client... found.

Checking for loopback interface... up.

Does tftp server work... yes.

Looking for a pxe daemon... which: no pxe in (/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin)

not found.

WARNING: your bootserver will be configured without a pxe daemon!

(ignore this warning if you're using a recent distro)

done.

What is the path to the pxelinux bootloader [/usr/lib/syslinux/pxelinux.0]?

Backing up /var/lib/tftpboot/pxelinux.cfg...

Moving /var/lib/tftpboot/pxelinux.cfg to /var/lib/tftpboot/pxelinux.cfg.si_mkbootserver.bak24...done.

Copying /var/lib/tftpboot/X86PC/UNDI/linux-install/pxelinux.cfg to /var/lib/tftpboot/pxelinux.cfg...Ok, configuration complete.

Once you're DHCP server is configured, you should be all set.

Do you want to run si_mkdhcpserver to configure your DHCP server ([y]/n)? y

Welcome to the SystemImager "si_mkdhcpserver" command. This command will

prepare this computer to be a DHCP server by creating a dhcpd.conf file

for use with your ISC DHCP server (v2 or v3).

If there is an existing file, it will be backed up with the
.beforesystemimager extension.

Continue? (y/[n]): y
Trying to probe your DNS domain. Please wait...

Type your response or hit <Enter> to accept [defaults]. If you don't
have a response, such as no first or second DNS server, just hit
<Enter> and none will be used.

What is your DHCP daemon major version number (2 or 3)? [3]:
What is the name of your DHCP daemon config file? [/etc/dhcpd.conf]:
What is your domain name? []: changzi.centos
What is your network number? [192.168.1.0]:
What is your netmask? [255.255.255.0]:
What is the starting IP address for your dhcp range? [192.168.1.1]: 192.168.1.50
What is the ending IP address for your dhcp range? [192.168.1.254]: 192.168.1.80
What is the IP address of your first DNS server? []:
What is the IP address of your default gateway? [192.168.1.254]: 192.168.1.1
What is the IP address of your image server? [192.168.1.254]: 192.168.1.63
What is the IP address of your boot server? [192.168.1.254]: 192.168.1.63
What is the IP address of your log server? []:
Use tmpfs staging on client? (If unsure, choose "n") [n]:
Do you want to use Flamethrower (multicast) to install your clients? [n]:

What... is the air-speed velocity of an unladen swallow? []:
Wrong!!! (with a Monty Python(TM) accent...)

Press <Enter> to continue...
Ahh, but seriously folks...

Here are the values you have chosen:

```
################################################################################
ISC DHCP daemon version:              3
ISC DHCP daemon config file:          /etc/dhcpd.conf
DNS domain name:                      changzi.centos
Network number:                       192.168.1.0
Netmask:                              255.255.255.0
Starting IP address for your DHCP range:  192.168.1.50
Ending IP address for your DHCP range:    192.168.1.80
First DNS server:
Second DNS server:
Third DNS server:
Default gateway:                      192.168.1.1
Image server:                         192.168.1.63
Boot server:                          192.168.1.63
Log server:
Log server port:
Flamethrower directory port:
Use tmpfs staging on client:          n
SSH files download URL:
################################################################################
```

Are you satisfied? (y/[n]): y

The dhcp server configuration file (/etc/dhcpd.conf) file has been

created for you. Please verify it for accuracy.

If this file does not look satisfactory, you can run this command again

to re-create it: "si_mkdhcpserver"

WARNING!: If you have multiple physical network interfaces, be sure to
edit the init script that starts dhcpd to specify the interface that
is connected to your DHCP clients. Here's an example:

Change "/usr/sbin/dhcpd" to "/usr/sbin/dhcpd eth1".

Depending on your distribution, you may be able to set this with the
"INTERFACES" variable in "/etc/default/dhcp", "/etc/default/dhcp3-server",
or similar, or in your dhcpd initialization script ("/etc/init.d/dhcpd",
"/etc/init.d/dhcp3-server", or similar).

Also, be sure to start or restart your dhcpd daemon. This can usually
be done with a command like "/etc/init.d/dhcpd restart" or similar.

Would you like me to restart your DHCP server software now? (y/[n]): y

关闭 dhcpd: [确定]

启动 dhcpd: [确定]

5，创建相应连接

如果镜像服务器中有多个镜像的话，就要告知哪一个节点安装哪一个镜像。si_addclients 为镜像的安装脚本创建符号链接。si_addclients 改写镜像服务器的/etc/hosts 和/var/lib/systemimager/scripts/hosts 文件。Hosts 文件为自动安装客户端查阅他们的主机名提供默认的机制。

在 si_addclients 的第一个配置部分，需要指出自动安装节点的主机名称样式。一个主机范围的字段和一个域名用来定义我们要自动安装的节点的主机名；在第二个配置部分，将前一部分定义的节点映射到到镜像；在第三个配置部分，si_addclients 命令请求 IP 地址范围，这个 IP 保存在/etc/hosts 和/var/lib/systemimager/scripts/hosts 文件中。当自动安装客户端启动时，它会从镜像服务器中检索后面的文件并应用它查找主机名。

si_addclients

Welcome to the SystemImager "si_addclients" utility

This utility has 3 sections.

"Section 1" will ask you for your hostname information.

"Section 2" will allow you to create softlinks from each client hostname to
your "master" script in the "/var/lib/systemimager/scripts" directory.

Example: www297.sh -> web_server_image_v1.master
"Section 3" will ask you for IP address information that will be combined
with the hostname information provided in Section 1 to create entries in
"/etc/hosts" for each of these same clients. New entries will be appended
to the end of "/etc/hosts". If you specify new hostnames for existing IP
addresses, those entries will be re-written in place to reflect the new
host names.

Continue? ([y]/n): y
si_addclients -- Section 1 (hostname information)

The next series of questions will be used to create a range of hostnames.
You will be asked for your domain name, the base host name, a beginning
number, and an ending number.

For example, if you answer:
domain name = systemimager.org
host range = www7-www11,www20

Then the result will be a series of hostnames that looks like this:
www7.systemimager.org
www8.systemimager.org
www9.systemimager.org
www10.systemimager.org
www11.systemimager.org
www20.systemimager.org

What is your domain name? []: changzi.centos

What is the hosts range that you want me to use? []: www50-www80

I will work with hostnames: www50-www80
in the domain: changzi.centos
Are you satisfied? (y/[n]): y
si_addclients -- Section 2 (soft links to master script)

Would you like me to create soft links to a "master" script so that hosts:
www50-www80
can be autoinstalled with one of the available images? ([y]/n): y
Here is a list of available autoinstall scripts:
backup
Which script would you like these hosts to be installed with?
[backup]:
Your soft links have been created.
Press <Enter> to continue...
si_addclients -- Section 3 (adding or modifying /etc/hosts entries)

Your target machines need to be able to determine their host names from their
IP addresses, unless their host name is specified in a local.cfg file.
The preferred method for doing this is with DNS. If you have a working DNS
that has IP address to hostname resolution properly configured for your
target machines, then answer "n" here.
If you don't have a working DNS, or you want to override the information in
DNS, then answer "y" here to add entries to the "/etc/hosts" file on your
image server. After adding these entries, the /etc/hosts file will be
copied to "/var/lib/systemimager/scripts" where it can be retrieved by your
target machines.
I will ask you for your clients' IP addresses one subnet at a time.
Would you like me to continue? (y/[n]): y
si_addclients -- Section 3 (adding or modifying /etc/hosts entries -- continued...)

Hostnames range is: www50-www80

What is the IPs address range (e.g. 10.0.0.1-10.0.0.100,10.0.0.101)?

[]:192.168.1.50-192.168.1.80

I will work with IP addresses: 192.168.1.50-192.168.1.80

and hostnames: www50-www80

Are you satisfied? (y/[n]): y

These entries have been added to /etc/hosts, and /etc/hosts has been copied

to /var/lib/systemimager/scripts for use by your auto-install clients.

Press <Enter> to continue...

si_addclients: successfully completed.

6，创建启动介质

见第 4 步骤中所描述的四种启动方式。此处所用的是从网络启动。使用以下命令，其它启动方法使用的命令详情见 systemimager 手册。

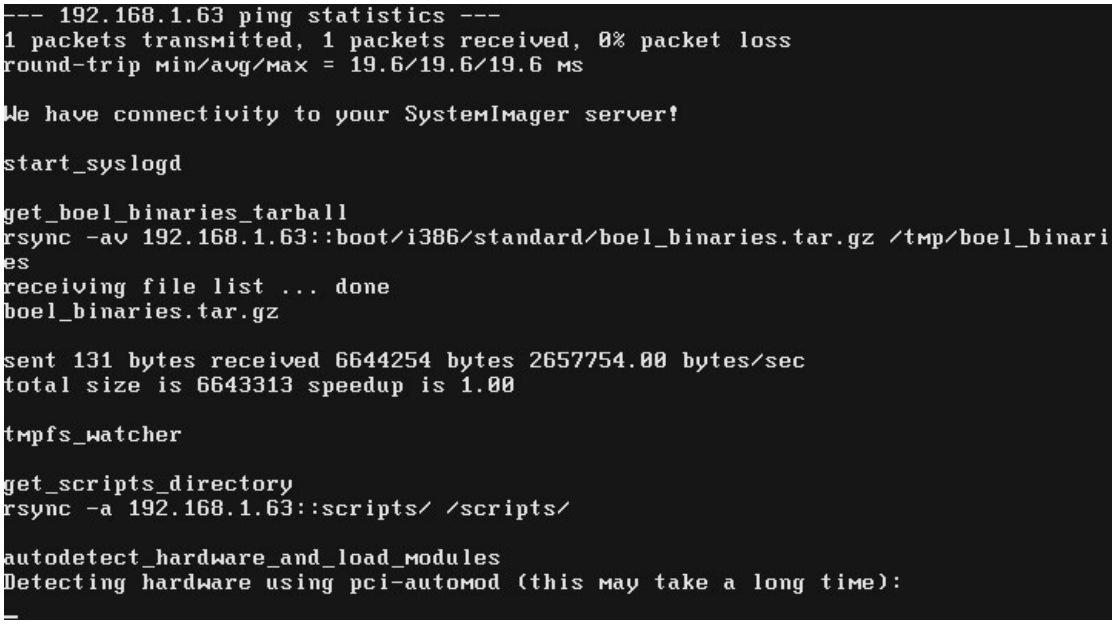
si_mkclientnetboot --netboot --clients www50-www80

[netboot] using the kernel and initrd.img for architecture: i386

[netboot] using the flavor: standard

7，启动节点安装镜像

最后就是启动节点，此处的节点均为虚拟机，从 DHCP 获得节点 IP，下图看到的 IP 为 79，因为装了两台，另一台的地址即为上面配置的 8 0 。开始自动安装，到了这一步是最令人欣喜的。



下载内核及 initrd

检测硬件

```
Adding 524280k swap on /dev/VolGroup00/LogVol01.  Priority:-1 extents:1 across:5
24280k
swapon /dev/VolGroup00/LogVol01 !! shellout
mke2fs -q -j /dev/VolGroup00/LogVol00 !! shellout
mkdir -p /a/ !! shellout
mount /dev/VolGroup00/LogVol00 /a/ -t ext3 -o defaults !! shellout
kjournald starting.  Commit interval 5 seconds
EXT3 FS on dm-0, internal journal
EXT3-fs: mounted filesystem with ordered data mode.
mke2fs -q -j /dev/sda1 !! shellout
tune2fs -L /boot /dev/sda1
tune2fs 1.38 (30-Jun-2005)
mkdir -p /a/boot !! shellout
mount /dev/sda1 /a/boot -t ext3 -o defaults !! shellout
kjournald starting.  Commit interval 5 seconds
EXT3 FS on sda1, internal journal
EXT3-fs: mounted filesystem with ordered data mode.
mkdir -p /a/proc !! shellout
mount proc /a/proc -t proc -o defaults !! shellout
mkdir -p /a/sys !! shellout
mount sysfs /a/sys -t sysfs -o defaults !! shellout
Quietly installing image...
/rsync -aHS --exclude=lost+found/ --exclude=/proc/* --numeric-ids 192.168.1.63::
backup/ /a/
_
```

开始传送文件

```
/boot/grub/grub.conf
/boot/grub/grub.conf
/etc/modprobe.conf
/etc/modules.conf
/etc/sysconfig/network
/etc/sysconfig/network-scripts/ifcfg-eth0

run_post_install_scripts
>>> 10all.fix_swap_uuids
>>> 95all.monitord_rebooted
>>> 99all.harmless_example_script

I live in /var/lib/systemimager/scripts/post-install.
See: /var/lib/systemimager/scripts/post-install/README for details.

umount /a/sys !! mount -no remount,ro /a//sys !! shellout
umount /a/proc !! mount -no remount,ro /a//proc !! shellout
umount /a/dev !! mount -no remount,ro /a//dev !! shellout
umount /a/boot !! mount -no remount,ro /a//boot !! shellout
umount /a/ !! mount -no remount,ro /a// !! shellout
Imaging completed
Terminated
I have been done for 1 seconds. Reboot me already!
I have been done for 2 seconds. Reboot me already!
_
```

传送完毕，等待重启。

重启系统，checking filesystems 后再次重启即可进入安装完成的系统中了。这一步用了１０分钟左右。同样，如果是实机的话性能会有所提高。为了进行比较一下，我用光盘安在虚拟机中安装了同样配置的系统，去掉启动后手工配置的时间是１３分钟。可见用这种方法部署系统还是有优势的。

8，其它

节点系统更新也是 SystemImager 的一个强大的功能，例如，想将 100 台服务器的内核更新，只需将样本节点的内核升级，获得样本节点的镜像，在其它节点系统中执行 si_updateclient，指定镜像后，便可以很快的与新的镜像同步，数据或是配置文件也可以用同样的方法分发。SystemImager 用来确保安全的生产部署,在更新镜像前保存当前生产环境的镜像，这样就提供了处理意外情况的机制。建立一个带有版本号的镜像池是一个很不错的选择，如果发现新的生产环境有问题，简单地用 si_updateclient 命令回滚到上次正常的生产镜像即可。

除了 SystemImager 这种工具可以执行自动化安装外，像 Red Hat 的 Kickstart 等，基于预定义的安装包清单来安装系统。然而，这种基于包的安装非常有限，因为它对非安装包的文件没有办法自动化安装，如果重新编译的内核，加入了一些非安装包的软件，或者是更改具体的配置文件，基于包安装的方法通常要求你写某些脚本或是编程来处理这些“特别案例”。相比之下，不如 SystemImager 使用起来方便快捷。在安全方面 SystemImager 还提供基于 OpenSSH 方式的安装。SystemImager 应用多种方法启动目标节点，将 Linux 操作系统和应用软件一次安装到位。采用分布式的网络传输结构和点对点的通信（SystemImager 提供用 BitTorrent 进行安装）方式可以有效缓解由网络带宽引起的性能瓶颈。

总结：这个实验做得不是很流畅，中间遇到好多问题，但大部份都 google 解决了。最近半个月以来一直在学习如何使用这个软件。主要的参考资料来源于 http://wiki.systemimager.org/index.php/Documentation，笔者下载并翻译了 SystemImager 手册，放到了博客里，但是对于这个软件所提供的更强大的功能尚未深入探究。

《SystemImager 手册》中提到过一些实际安装的性能：

Ole Holm Nielsen，物理系，丹麦技术大学报告：

在我们用 SystemImager 安装时，可以在 6 分钟内安装 1.8G 的镜像到 18 个客户端。请见 The NIFLHEIM SystemImager Page(http://www.fysik.dtu.dk/CAMP/Niflheim/systemimager.html)。我们的服务器拥有 Gigabit 网络，2GB 的 RAM，dual Intel Xeon 2.4 GHz，客户端是 Intel P4 和 100 Mbit 网络。

James Braid 报告： 从一个 Celeron 700/512Mb 服务器，100Mbit 网络，我们做到了 7～10 分钟安装大概 1G 的镜像。硬盘是设置了 LVM 的 5x 120Gb Seagate Barracuda V（non striped），文件系统为 ReiserFS。

参考：

[1] http://wiki.systemimager.org/index.php/Quick_Start_HOWTO

[2] http://www.howtoforge.com/howto_linux_systemimager

PS：有位朋友反映通过 PXE 安装（用虚拟机），启动后，到读取 master 脚本时卡住不动了。就是在检测硬件这一步。如何造成的这个问题我现在也搞不清，是没有想考虑，呵呵，此处建议使用虚拟机 VM6.5 的版本，因为那位朋友换这个版本后就好使了，我使用的也是这个版本，其中并没有特殊设置，正常建立一个虚拟机即可。当然，由于不具备实机的实验环境，所以不知道在实机中使用 SystemImager 会遇到什么问题。

Apache 升级到了 nginx 的几个注意点

最近把整站从 apache 升级到了 nginx，客户的站点大概有 30 台服务器大部分架构位 tomcat+apache，只有一个 php 页面。

以下是我升级遇到的几个问题的注意点：

1、当我们去访问服务器上的一个目录时候，他不会自动加上一个/ ，浏览器会给出改页无法打开的错误，这个时候浏览器去取的地址实际上是 upstream 中所写的地址和端口或如果没有使用 upstream 时 当使用 localhost 做 servername 时候 浏览器会去访问 http://127.0.0.1/dir。

解决办法：

在每个虚拟主机的 server 定义中加上

```
if (-d $request_filename) {
            rewrite ^/(.*)([^/])$ http://$host/$1$2/ permanent;
        }
```

注意 root 字段的定义也一定要出现在 server 中 如果 server 中没有定义 root 错误还将存在

例子：

```
server {
        listen          800;
        server_name www.1.com;
        root   /opt/1-index;   //这边定义了 就会在目录访问的时候加上/  如果这边没有定义这个上面的 url 重写依然不会生效
        include         vhost/alias.conf;
        include         vhost/proxy.conf;
        if (-d $request_filename) {
            rewrite ^/(.*)([^/])$ http://$host/$1$2/ permanent;
        }
        error_page   405 =200 @405;
        location @405 {
        proxy_pass     http://PROXY_STATIC;
        }
        location / {
            root   /opt/1-index;      //只在这边定义是没有用的，这边甚至可以不做定义
            rewrite ^/(\d+)\.home$ /index.html?userId=$1 last;
            index  index.html index.htm;
        }
```

2、url 重写的注意事项：

原有的 url 支持正则 重写的 url 不支持正则

```
rewrite ^/(\d+)\.home$ /index.html?userId=$1 last;
```

这个重写中 ^/(\d+)\.home$ 这部分支持正则

而/index.html?userId=$1

不要用正则 也不匹配正则 /index.html?*userId=$1 这样他就会去找.html?*userId=$1 这个 url 然后给你个 404 not found

3、post 方式去访问静态文件

Apache、IIS、Nginx 等绝大多数 web 服务器，都不允许静态文件响应 POST 请求，否则会返回“HTTP/1.1 405 Method not allowed”错误。（但是之前程序在 apache 上跑 没问题）
如果有这个需求呢 就要做如下配置了

```
error_page   405 =200 @405;
        location @405 {
        proxy_pass     http://PROXY_STATIC;
    #      root /usr/local/nginx/html;
        }
```

把所有 405 错误重定向成 200 然后吧所有 405 错误的请求全部交给一个代理去执行
或者写上本地路径,因为我的路径比较多 所以重定向请求到一台 web 服务器上了

4、关于防盗链

1.com 的需求是不是从本来来的请求给除一个 403
因为在虚拟主机里配置毫无作用 可能是我们用的是虚拟目录的缘故
所以我们直接对目录做的防盗链
在 alias 里
别名配置

```
        location /res/ {
        alias   /opt/Src/;
        valid_referers none blocked server_names *.1.com ;
        if ($invalid_referer) {
    return 403;
        }
    }
```

5、关于动态请求转发

```
location  ~ ^/login/(.*\.do)$ {
        proxy_pass     http://login ;
    proxy_set_header  X-Real-IP  $remote_addr;
}
```

~ ^/login/(.*\.do)$ 这个表示 凡是匹配/login/ 下 .do 的都转发到一个 upstream 池里处理 这里的$符号并不起多大作用 只要是有.do 的他会全部转 并不是以.do 结尾的才转

6、关于 php 上传文件大小的问题

只改 php 里的配置是没有用的，需要更改的地方还有 nginx 的配置：

```
client_max_body_size 10M;
```

他的默认值是 1M;

以上就是我要提到的几个基本的注意点 。

Linux 系统下 Bugzilla for Oracle 安装笔记

ChinaUnix 网友：bugzilla-orcl

Bugzilla3.2 今天发布，该版本增加很多新的功能，其中重要的新功能包括 Oracle 数据库支持. 下面介绍 Bugzilla + Oracle + Linux 的安装过程笔记。

== 准备工作 ==
检查所需软件:
Perl 5.81 及以上
Oracle v10.02.0 及以上
Apache(httpd),sendmail 等

== Oracle 的安装配置 ==

Oracle 需要 v10.02.0 以上版本支持，可以是企业版，也可以是 XE 版，具体用哪个版本，可以根据需要来确定.

Oracle 安装文档到处都有，这里就不说了，开始说 Oracle 的配置吧.

=== 1. 创建 Tablespace(可选):===
用 sys 或者 system 用户登陆 sqlplus, 执行:

```
CREATE TABLESPACE bugs
   DATAFILE '/u01/oradata/bugzilla.dbf' SIZE 500M
   AUTOEXTEND ON NEXT 30M MAXSIZE UNLIMITED;
```

=== 2. 创建 bugzilla 用户: ===
用 sys 或者 system 用户登陆 sqlplus, 执行(创建数据库用户 bugs, 密码 bugs):

```
CREATE USER bugs
   IDENTIFIED BY "bugs"
   DEFAULT TABLESPACE bugs
   TEMPORARY TABLESPACE TEMP
   PROFILE DEFAULT;
   -- GRANT/REVOKE ROLE PRIVILEGES
   GRANT CONNECT TO bugs;
   GRANT RESOURCE TO bugs;
   -- GRANT/REVOKE SYSTEM PRIVILEGES
   GRANT UNLIMITED TABLESPACE TO bugs;
   GRANT EXECUTE ON CTXSYS.CTX_DDL TO bugs;
```

=== Apache 配置 ===
配置 ORACLE_HOME 和 LD_LIBRARY_PATH, 例如:

vi /etc/http/conf.d/bugzilla.conf

输入保存:
Set ORACLE_HOME and LD_LIBRARY_PATH
SetEnv ORACLE_HOME /usr/lib/oracle/xe/app/oracle/product/10.2.0/server
SetEnv LD_LIBRARY_PATH /usr/lib/oracle/xe/app/oracle/product/10.2.0/server/lib

== 安装 Bugzilla 3.2 ==
=== 下载并解压 ===
下载 Bugzilla3.2, http://ftp.mozilla.org/pub/mozilla.org/webtools/bugzilla-3.2.tar.gz
cd /var/www/html/
wget http://ftp.mozilla.org/pub/mozilla.org/webtools/bugzilla-3.2.tar.gz
tar zxvf bugzilla-3.2.tar.gz
mv bugzilla-3.2 bugzilla

=== checksetup.pl ===
接下来执行 checksetup.pl
cd bugzilla
./checksetup.pl

根据提示安装必须的 perl 模块, 在此之前你需要安装 perl-CPAN
yum install perl-CPAN
Linux 下面安装 Bugzilla 所需的 perl 模块，最简单的方式是利用 Bugzilla 提供的 install-module.pl 工具, 执行:
/usr/bin/perl install-module.pl --all
遇到所有提示，默认即可

安装 DBD::Oracle

export ORACLE_HOME="/usr/lib/oracle/xe/app/oracle/product/10.2.0/server"
export LD_LIBRARY_PATH="/usr/lib/oracle/xe/app/oracle/product/10.2.0/server/lib"
/usr/bin/perl install-module.pl DBD::Oracle

"说明: 如果 Oracle 数据库是在远程，则安装 Bugzilla 的机器上面必须安装 Oracle Instant Client 或者 Oracle Client，然后设置相应的 ORACLE_HOME 和 LD_LIBRARY_PATH"

再次执行 checksetup.pl, 产生 localconfig 文件:
./checksetup.pl

编辑 localconfig 文件:

$db_driver = 'oracle';
$db_host = '10.182.120.189'; <-- 数据库 IP
$db_name = 'XE'; <-- 数据库实例名 SID

$db_user = 'bugs'; <-- 数据库用户名
$db_pass = 'bugs' <-- 数据库密码

配置完 localconfig 之后，再次运行 checksetup.pl:

./checksetup.pl
填入管理员信息:

Enter the e-mail address of the administrator: admin@kk.com
Enter the real name of the administrator: Admin
Enter a password for the administrator account:
Please retype the password to verify:

=== 配置 Apache ===
配置/etc/httpd/conf.d/bugzilla.conf, 加入

```
<Directory /var/www/html/bugzilla>
   AddHandler cgi-script .cgi
   Options +Indexes +ExecCGI
   DirectoryIndex index.cgi
   AllowOverride Limit
</Directory>
```

重新启动 Apache
service httpd restart

用管理员登陆 http://yourip/bugzilla/, 然后配置 urlbase

安装完成!

"说明：如果遇到类似下面的问题, 请关闭系统的 selinux，然后重启系统："
'oracle' is not a valid choice for $db_driver in localconfig: Can't load 'lib/i386-linux-thread-
 multi/auto/DBD/Oracle/Oracle.so' for module DBD::Oracle:
libclntsh.so.10.1: cannot enable executable stack as shared object requires: Permission
denied at
 /usr/lib/perl5/5.10.0/i386-linux-thread-
multi/DynaLoader.pm line 203.
at Bugzilla/DB/Oracle.pm line 41
Compilation failed in require at Bugzilla/DB/Oracle.pm line 41.
BEGIN failed--compilation aborted at Bugzilla/DB/Oracle.pm line 41.
Compilation failed in require at (eval 30) line 3.

网友热评

热点技术评论

ls 如何只显示目录信息

Linux 内核中的红黑树

关于 kernel patch 的提交

在计算机领域做研究的一些想法

grub>这个是什么意思?

嵌入式操作系统的成功之道

今天终天在虚拟机上做好 LFS 系统!

个人对 kobject 的一点研究

vsftpd 设置小问题

初学 linux，选择哪个版本安装比较好?

如何对 ssh 做一些相关设置呢?

Debian 4.0 启动速度怎么这么慢?

VM 中的两台 linux 互访的问题

请教一个 iptables --set-mark 的含义

iptables 映射两次端口后的问题

为什么 top 中只有一个 running 的进程?

C++中 catch 引用要比 catch 对象效率低!

如何用 sed 删除每行末尾的 n 个字符?

向高手们求解大文件记录更新问题

RS、ORS、FS、OFS 相关问题

awk $*是什么意思?

关于编译环境方面的问题

FreeBSD 7.1 GNOME liveCD 中文桌面版发布!

热点新闻评论

失业感言，学 linux 的都来看看

有关三流大学毕业生职业瓶颈的思考?

什么在阻碍您使用 Linux/BSD/Solaris?

09 第一问：龙芯 08 有多少成绩？！

2009 年绝不可能发生的 10 大 IT 事件

金融数据大集中带给 DBA 的新机遇

计算机小硕的迷茫，求指点

集群式超级计算机

41 年后 90%语言将消失 互联网加速语言灭绝

网友感叹：Linux 光盘在市场上消失?

有可能在 2009 年消失的 IT 企业

我的分析:sun 什么时候会倒掉?

点评 Linux 难称完美的几大命门

台湾公务单位倡导封杀 Office 2007 文件

微软 Zunes 播放器死机 源于开放源码严重 bug

妇女称 Ubuntu 不好用 惹恼 Linux 用户

三款 Linux 操作系统发展大剖析

开发人员希望使用的六种脚本语言

气愤，技嘉歧视 Linux 用户！！

ccnp 做企业网管合适吗?

评《Windows 7 无法击败 Ubuntu 的 7 个原因》

C++的爱好者对 D 语言感兴趣吗?

7.1 让我重新燃起用 FB 当桌面的希望之火